Capstone Project: Full VAPT Cycle

Activities

Tools: Kali Linux, Metasploit, OpenVAS, Google Docs

Tasks: Simulate full penetration testing cycle, exploit vulnerabilities, document and report findings.

Enhanced Tasks

Target: DVWA (Damn Vulnerable Web Application)

Vulnerability: SQL Injection

Tool Used: sqlmap

Command:

```
sqlmap -u
"http://172.17.0.2/vulnerabilities/sqli/?id=1&Submit=Submit#"
--cookie="PHPSESSID=xyz; security=low" –dump
```

Methodology (TryHackMe-Style Exploitation Workflow)

1. Identify the injectable parameter

- Navigate to DVWA > SQL Injection module.
- Enter a test input such as:
  `1'`
- Error messages or unexpected output confirm SQL Injection vulnerability.
- Note the vulnerable parameter: id.

  Module: /vulnerabilities/sqli/
  Parameter: id

2. Use sqlmap to enumerate the database

Run sqlmap with authentication cookies (DVWA requires login):

```
sqlmap -u
"http://172.17.0.2/vulnerabilities/sqli/?id=1&Submit=Submit#"
--cookie="PHPSESSID=xyz; security=low" --dbs
```

This enumerates the backend database names (e.g., dvwa, information_schema).

3. Dump database tables

Once the DVWA database is identified:

```
sqlmap -u
"http://172.17.0.2/vulnerabilities/sqli/?id=1&Submit=Submit#"
--cookie="PHPSESSID=xyz; security=low" -D dvwa –tables
```

Then dump sensitive tables:

```
sqlmap -u
"http://172.17.0.2/vulnerabilities/sqli/?id=1&Submit=Submit#"
--cookie="PHPSESSID=xyz; security=low" -D dvwa -T users –dump
```

2. Stored XSS (Medium Severity)
Target: DVWA → vulnerabilities/xss_s/
Parameter: message
Method Used: Manual injection (TryHackMe methodology)
Module: /vulnerabilities/xss_s/
Payload Tested:

```
<script>alert('Vulnerable XSS');</script>
```

Findings:

- Input was stored in the backend database.
- Payload executed every time the page was viewed.
- No input sanitization or server-side validation.
- Stored XSS allows:
    - Cookie theft
    - Session hijacking
    - Credential compromise
    - Browser-based attacks

Reflected XSS (Low–Medium Severity)
Target: DVWA → vulnerabilities/xss_r/
Method: Manual testing + TryHackMe technique
Module: /vulnerabilities/xss_r/
Payload:

```
"><script>alert('reflected')</script>
```

Findings:

- Parameter is echoed back to the page without escaping.
- Allows one-time theft of:
    - Session ID
    - Tokens
    - Cookies

Detection (OpenVAS Scan Findings)
OpenVAS was run against the DVWA host on port 80 to detect server-level issues.
Key Findings from OpenVAS:

- Missing 'HttpOnly' Cookie Attribute (Medium)
- Cleartext Transmission over HTTP (Medium)
- Backup File Disclosure (Medium)
- ICMP Timestamp Reply (Low)
- TCP Timestamp Disclosure (Low)

These findings relate to **misconfigurations and HTTP security weaknesses**, not application logic vulnerabilities.

OpenVAS scan performed against **192.168.0.13**, port 80.

Vulnerability	Severity	Host	Location	Timestamp
Operating System (OS) End of Life Detection	High	192.168.0.13	general/tcp	2025-11-20 19:43 UTC
Missing ‘HttpOnly’ Cookie Attribute	Medium	192.168.0.13	80/tcp	2025-11-20 19:45 UTC
Backup File Scanner (HTTP)	Medium	192.168.0.13	80/tcp	2025-11-20 19:46 UTC
Cleartext Transmission via HTTP	Medium	192.168.0.13	80/tcp	2025-11-20 19:44 UTC
TCP Timestamp Information Disclosure	Low	192.168.0.13	general/tcp	2025-11-20 19:43 UTC
ICMP Timestamp Reply Disclosure	Low	192.168.0.13	general/icmp	2025-11-20 19:43 UTC

Combined PTES Activity Log (Manual Exploitation + OpenVAS Detection)

Timestamp	Target IP	Vulnerability	PTES Phase
2025-08-18 12:00:00	192.168.0.13	SQL Injection (Manual – sqlmap)	Exploitation
2025-08-18 12:05:00	192.168.0.13	Stored XSS (Manual)	Exploitation
2025-08-18 12:08:00	192.168.0.13	Reflected XSS (Manual)	Exploitation
2025-11-20 19:43:00	192.168.0.13	OS End of Life Detection (OpenVAS)	Vulnerability Analysis
2025-11-20 19:45:00	192.168.0.13	Missing HttpOnly Cookie Attribute (OpenVAS)	Vulnerability Analysis
2025-11-20 19:46:00	192.168.0.13	Backup File Scanner – Sensitive Data Exposure	Vulnerability Analysis
2025-11-20 19:44:00	192.168.0.13	Cleartext Transmission via HTTP (OpenVAS)	Vulnerability Analysis
2025-11-20 19:43:00	192.168.0.13	TCP Timestamp Disclosure (OpenVAS)	Discovery
2025-11-20 19:43:00	192.168.0.13	ICMP Timestamp Reply Disclosure (OpenVAS)	Discovery

Remediation Suggestions

- Apply input validation and parameterized queries (fix SQLi).
- Implement output encoding and filtering (fix XSS).
- Enable HTTPS and enforce TLS 1.2/1.3 (fix cleartext HTTP).
- Set secure cookie flags: HttpOnly, SameSite, Secure.
- Remove backup files from web root.
- Update OS and web server to a supported version.
- Disable ICMP timestamps and reduce information leakage.

PTES Report

This particular test is executed on guidelines provided by Penetration Testing Execution Standard (PTES). This begins with intelligence-gathering processes involving active host scanning, scanning of ports, active services scanning, as well as scanning for vulnerabilities conducted through open-source techniques. This is followed by threat modeling processes for identifying high-risk areas to be targeted by attacks.

Vulnerability analysis scanners include Nmap, OpenVAS, and Nikto, which have been used to identify outdated versions of software installed on web servers, vulnerabilities linked to improper software configuration, and the lack of access control measures while identifying vulnerabilities for this project. The analysis of exploitation attacks involved attacks to determine their exploitability for authentication bypass vulnerabilities, directory traversal vulnerabilities, and improper configuration of services.

In the post-exploitation phase, it was determined that the attacker could move around within the network for privilege escalation and access to sensitive information. It is also important to note here that all findings have been documented properly to facilitate activities for remediation such as patching vulnerabilities within software components, locking down configurations for increased security measures, enforcing strong authentication policies, and disabling unwanted services.

This helped to create a safe and realistic assessment of the security readiness of this system because it ensured conformance to PTES:

Non-Technical Summary

A security check was done to find out how safe the system is from possible cyberattacks. First, we viewed the system from the outside-in to understand what information was publicly visible. Then, the scanning tools were applied to detect weaknesses such as outdated software or unsafe settings. Thirdly, controlled tests were conducted to see whether an attacker could misuse these weaknesses. Verification of findings was done and documented in simple, clear fixes. Overall, the assessment shows where the system is strong and where improvements are needed to better reduce risks and protect data.