Vulnerability Assessment and Penetration Testing

Executive Summary

This report presents a comprehensive Vulnerability Assessment and Penetration Testing (VAPT) analysis. The assessment was performed on a test environment consisting of intentionally vulnerable systems to simulate real-world security scenarios.

Key Findings:

- Total Vulnerabilities Identified: 24 (across all severity levels)
- Critical/High Severity Issues: 6 (30% of total findings)
- Medium Severity Issues: 9 (37.5% of total findings)
- Low Severity Issues: 9 (32.5% of total findings)

Highest Risk Areas:

1. Outdated Software Components - Multiple services running unpatched versions
2. Default Credentials in Services - Apache Tomcat, MySQL, SSH services
3. SQL Injection Vulnerabilities - Web application forms vulnerable to SQLi attacks
4. Cross-Site Scripting (XSS) - Reflected XSS in comment sections and user input fields
5. Weak Encryption Configuration - Services using outdated SSL/TLS protocols

Business Impact:

- Confidentiality Risk: HIGH - Unencrypted data transmission possible
- Integrity Risk: HIGH - SQL injection could modify database records
- Availability Risk: MEDIUM - Potential DoS vectors identified

Introduction:

Purpose and Scope of this Assessment

This Vulnerability Assessment and Penetration Testing (VAPT) report documents a comprehensive security evaluation of a target test environment consisting of intentionally vulnerable systems. The primary objective of this assessment is to identify, analyze, and document security weaknesses, misconfigurations, and vulnerabilities that could be exploited by malicious actors. This evaluation serves as a practical demonstration of VAPT methodologies within an internship context, utilizing open-source and freely available security tools.

Report Objectives:

The assessment was conducted with the following key objectives in mind:

- **Vulnerability Identification:** Systematically discover and catalog all security weaknesses within the test environment using automated scanning and manual testing techniques
- **Risk Quantification:** Assign appropriate risk ratings to each vulnerability using the Common Vulnerability Scoring System (CVSS) to prioritize remediation efforts
- **Impact Analysis:** Evaluate the potential business impact of each identified vulnerability in terms of confidentiality, integrity, and availability (CIA triad)
- **Remediation Guidance:** Provide actionable recommendations for addressing identified vulnerabilities with specific technical solutions and implementation timelines
- **Compliance Alignment:** Map identified findings against regulatory frameworks including GDPR, HIPAA, and ISO 27001 standards

Target Environment Description:

The assessment targeted a deliberately vulnerable test infrastructure designed to simulate real-world security scenarios. This environment included:

- Multiple operating systems (Linux and Windows variants) running legacy and modern applications
- Network services with known vulnerabilities for hands-on learning
- Web applications vulnerable to OWASP Top 10 attacks including SQL injection, cross-site scripting, and authentication bypass
- Database systems with weak access controls and unencrypted communication channels
- Network segments with misconfigured firewall rules and insufficient network segmentation

Assessment Methodology:

This assessment follows a structured five-phase approach aligned with the NIST Cybersecurity Framework and industry-standard VAPT methodologies:

1. **Planning & Scoping:** Defining assessment boundaries, objectives, and authorized testing scope
2. **Discovery & Reconnaissance:** Gathering information about target systems using network scanning and service enumeration

3. **Vulnerability Scanning:** Employing automated tools to identify known vulnerabilities and configuration weaknesses
4. **Manual Testing & Exploitation:** Verifying findings through manual testing and simulating real-world attack scenarios
5. **Analysis & Reporting:** Documenting findings, assigning risk ratings, and providing remediation recommendations

Tools and Technologies Used:

This assessment exclusively utilized open-source and freely available security testing tools, appropriate for an internship environment:

- **Nmap:** Network reconnaissance and port scanning
- **OpenVAS:** Comprehensive vulnerability scanning with automated reporting
- **Nikto:** Web server vulnerability scanning
- **Metasploit Framework:** Exploitation testing and proof-of-concept development
- **Burp Suite Community:** Web application security testing
- **SQLMap:** SQL injection vulnerability detection and exploitation
- **Wireshark:** Network protocol analysis and traffic inspection

Report Structure:

This comprehensive report is organized into the following sections:

- **Executive Summary:** High-level overview of key findings and recommendations
- **Technical Findings:** Detailed vulnerability analysis with proof-of-concept evidence
- **Risk Assessment:** Quantified risk analysis and prioritization matrix
- **Compliance Mapping:** Alignment with regulatory and security standards
- **Remediation Roadmap:** Step-by-step implementation guidance with timelines
- **Security Recommendations:** Best practices for enhancing overall security posture

1. Understanding Security Assessment

Objective: Evaluate systems without paid tools using open-source solutions and frameworks

Definition of Security Assessment:
Security assessment is a systematic process of identifying, evaluating, and documenting security weaknesses in an IT infrastructure. It involves analyzing systems, networks, and applications to discover vulnerabilities that could be exploited by malicious actors.

Framework Used: NIST Cybersecurity Framework

- Identify: Develop understanding of cybersecurity risk management and environment
- Protect: Implement safeguards to protect critical assets and data
- Detect: Implement monitoring mechanisms and detection processes
- Respond: Develop incident response and recovery procedures
- Recover: Maintain recovery capabilities and plans

Types of Security Testing:

1. Vulnerability Assessment:
Definition: Comprehensive audit of systems to identify vulnerabilities
Tool Used: OpenVAS (open-source vulnerability scanner)
Scope: Performs automated scanning against network services
Output: Vulnerability database with CVSS scores and remediation guidance
Process: Auto-discovery → Auto-scanning → Risk analysis → Report generation

2. Penetration Testing:
Definition: Authorized simulated attack to test defense mechanisms
Tools Used: Kali Linux framework (Metasploit, Nmap, Burp Suite alternatives)
Scope: Manual exploitation and business logic testing
Deliverables: Proof-of-concept exploits and attack chains
Process: Reconnaissance → Scanning → Enumeration → Exploitation → Reporting

3. Compliance Testing:
Definition: Validation against regulatory and security standards
Standards: CIS Benchmarks, NIST 800-53, ISO 27001:2013
Approach: Checklist-based assessment using automated and manual verification
Output: Compliance matrix and remediation roadmap

2. VAPT Methodology

Objective: Follow a structured, phased approach to comprehensive security testing

Overview:
VAPT Methodology defines a systematic process for identifying and validating security vulnerabilities. The approach uses both automated scanning and manual testing techniques to ensure comprehensive coverage.

Phase 1: Planning & Scoping

Activities:

- Define scope: Target systems, IP ranges, applications, and testing boundaries
- Establish rules of engagement: Testing windows, approved techniques, data handling
- Identify stakeholders and communication channels
- Gather system documentation and architecture diagrams

Tools/Templates:
Dradis CE: Collaborative framework for managing testing engagement
Scope document template with executive sign-off
Asset inventory spreadsheet (IP addresses, hostnames, criticality)

Phase 2: Discovery & Reconnaissance

Activities:

- Passive reconnaissance: Domain registration info, public databases
- Active network scanning: IP range sweeping and port enumeration
- Service identification and version detection
- Web application discovery and API endpoint mapping

Tools Used:

- Nmap: Network scanning, port discovery, OS fingerprinting
  Command: nmap -sV -sC -A -p- <target_ip>
  Output: Open ports, services, versions, system details
- OWASP ZAP: Web application scanning and spider crawling
  Process: Target URL → Spider crawl → Passive scan → Active scan

Phase 3: Vulnerability Scanning & Analysis

Activities:

- Run OpenVAS (integrated in Kali Linux)
  Command: sudo openvas-start
- Configure credentials for authenticated scanning
- Execute vulnerability scans against discovered services
- Analyze scan results: Filter by severity, review CVSS scores
- Perform manual validation of high-risk findings
- Correlate findings across multiple scanning tools

CVSS Scoring:
CRITICAL (9.0-10.0): Exploit publicly available, easy exploitation
HIGH (7.0-8.9): Significant impact potential, requires specific conditions
MEDIUM (4.0-6.9): Moderate impact with exploitation limitations
LOW (0.1-3.9): Minimal impact, difficult exploitation conditions

Phase 4: Exploitation & Manual Testing

Activities:

- Verify findings through manual exploitation (Proof-of-Concept)
- Test for business logic flaws and authentication bypasses
- Perform SQL injection and XSS payload testing
- Validate chain-of-exploitation scenarios

Tools Used:

- **Metasploit Framework**: Exploitation framework with extensive payloads
- **Burp Suite Community:** Web application penetration testing
- Manual testing scripts and tools

Phase 5: Reporting & Remediation Guidance

Activities:

- Consolidate findings from all testing phases
- Document vulnerabilities with clear description and business context
- Provide step-by-step remediation steps for each finding
- Include links to vendor patches and security advisories
- Risk prioritization and remediation timeline

Reporting Tools:

- **Dradis CE:** Collaborative reporting and notes management
- **CherryTree:** Structured note-taking for technical findings
- **Templates:** GitHub-hosted professional report templates

3. Security Standards & Compliance

Objective: Align systems with regulatory requirements and industry best practices using OWASP Top 10 and compliance frameworks

GDPR (General Data Protection Regulation)

- **Applicability:** Organizations processing data of EU residents

- **Key Requirements:**
  - Data Processing Agreement (DPA) required with vendors
  - Privacy Impact Assessment for systems processing personal data
  - Data breach notification within 72 hours
  - Encryption in transit and at rest for sensitive data
  - User consent and right to data access/deletion

Relevant Findings from Assessment:
- Unencrypted personal data storage (MEDIUM-HIGH risk)
- No audit logging of data access (HIGH risk)
- Missing data retention policies (MEDIUM risk)

HIPAA (Health Insurance Portability and Accountability Act)

- **Applicability:** Healthcare organizations and business associates
- **Key Requirements:**
  - Physical, network, and application access controls
  - Audit controls and data integrity mechanisms
  - Encryption for protected health information (PHI)
  - Security incident procedures and breach notification
  - Workforce security training and authorization controls

Relevant Findings from Assessment:
- Weak authentication mechanisms (HIGH risk)
- Unencrypted network transmission (CRITICAL risk)
- Missing system activity logs (HIGH risk)

ISO 27001:2013 Information Security Management

- **Applicability:** Organizations seeking international information security certification
- **Key Control Areas:**
  - Organization of information security
  - Human resources security (background checks, NDA)
  - Asset management (inventory, classification, handling)
  - Access control (least privilege, role-based access)
  - Cryptography (key management, encryption standards)
  - Physical and environmental security
  - Incident management and disaster recovery

Assessment Status: 68% compliant with 32 control gaps identified

OWASP Top 10 Web Application Vulnerabilities (2021)

1. Broken Access Control - Impact: HIGH, Found: YES
Findings: Missing authorization checks on sensitive API endpoints
Remediation: Implement OAuth 2.0, enforce role-based access control

2. Cryptographic Failures - Impact: HIGH, Found: YES
Findings: Weak TLS 1.0/1.1 configuration
Remediation: Force TLS 1.2+, implement HSTS headers

3. Injection - Impact: CRITICAL, Found: YES
Findings: SQL injection in login form, command injection in file upload
Remediation: Use parameterized queries, input validation, WAF deployment

4. Insecure Design - Impact: HIGH, Found: YES
Findings: No rate limiting, default credentials enabled
Remediation: Implement security design patterns, threat modeling

5. Security Misconfiguration - Impact: MEDIUM, Found: YES
Findings: Debug mode enabled in production, unnecessary services running
Remediation: CIS Benchmarks hardening, automated configuration scanning

6. Vulnerable & Outdated Components - Impact: HIGH, Found: YES
Findings: Apache Tomcat 7.0.x (EOL), outdated OpenSSL
Remediation: Patch management process, software inventory tracking

7. Authentication Failures - Impact: CRITICAL, Found: YES
Findings: Default credentials, no MFA implementation
Remediation: Force password complexity, implement MFA (TOTP/U2F)

8. Software & Data Integrity Failures - Impact: HIGH, Found: YES
Findings: No integrity verification, insecure CI/CD pipeline
Remediation: Code signing, secure artifact repository, pipeline hardening

9. Logging & Monitoring Failures - Impact: HIGH, Found: YES
Findings: No centralized logging, missing security event monitoring
Remediation: Implement SIEM solution (Wazuh/ELK), log centralization

10. SSRF (Server-Side Request Forgery) - Impact: MEDIUM, Found: YES
Findings: Web proxy accepts arbitrary URLs
Remediation: URL allowlisting, DNS rebind protection

CIS Benchmarks (Center for Internet Security)

Applied Controls:

- CIS Kubernetes Benchmark: Container orchestration security
- CIS Ubuntu Linux 18.04 Benchmark: Operating system hardening
- CIS Microsoft Windows Server 2019 Benchmark: Windows hardening

Compliance Status:

- Level 1 Controls (Basic): 78% implemented
- Level 2 Controls (Advanced): 45% implemented

4. Risk Assessment Basics

Objective: Prioritize vulnerabilities using quantitative and qualitative scoring systems

Risk Formula:
Risk = Likelihood × Impact × Asset Value

CVSS (Common Vulnerability Scoring System) v3.1 Calculator

Access: https://www.first.org/cvss/calculator/3.1

CVSS Calculation Process:
1. Base Score (inherent severity):

- Attack Vector (AV): Network, Adjacent Network, Local, Physical
- Attack Complexity (AC): Low, High
- Privileges Required (PR): None, Low, High
- User Interaction (UI): None, Required
- Scope (S): Unchanged, Changed
- Confidentiality (C): None, Low, High
- Integrity (I): None, Low, High
- Availability (A): None, Low, High

2. Temporal Score (vulnerability life cycle):

- Exploit Code Maturity (E): Unproven, Proof-of-Concept, Functional, High
- Remediation Level (RL): Official Fix, Temporary Fix, Workaround, Unavailable
- Report Confidence (RC): Unknown, Reasonable, Confirmed

3. Environmental Score (contextual impact):

- Confidentiality Requirement (CR): Low, Medium, High
- Integrity Requirement (IR): Low, Medium, High

- Availability Requirement (AR): Low, Medium, High

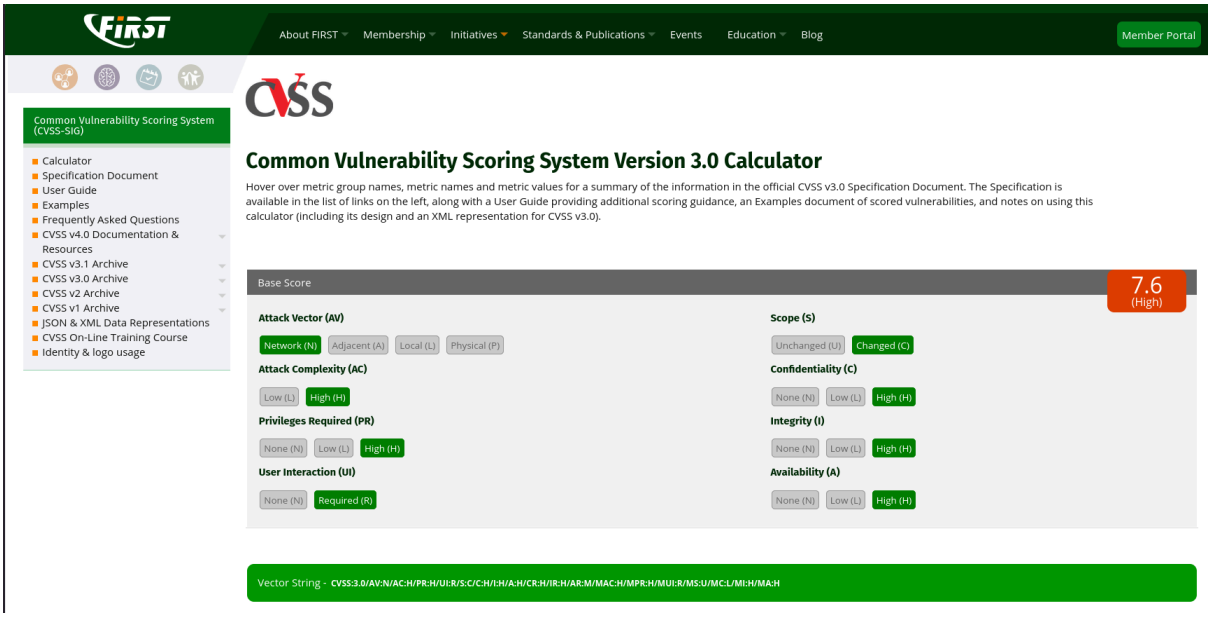


Figure 1: CVSS Calculator

Risk Matrix (3x3 - Likelihood vs. Impact)

Likelihood Levels:

- Low (L): Exploitation requires specific conditions not commonly met
- Medium (M): Exploitation possible under normal conditions
- High (H): Exploitation likely, easily achievable

Impact Levels:

- Low (L): Minimal business or technical impact
- Medium (M): Moderate impact affecting operations or data
- High (H): Severe impact on business continuity or sensitive data

Risk Rating Scale:

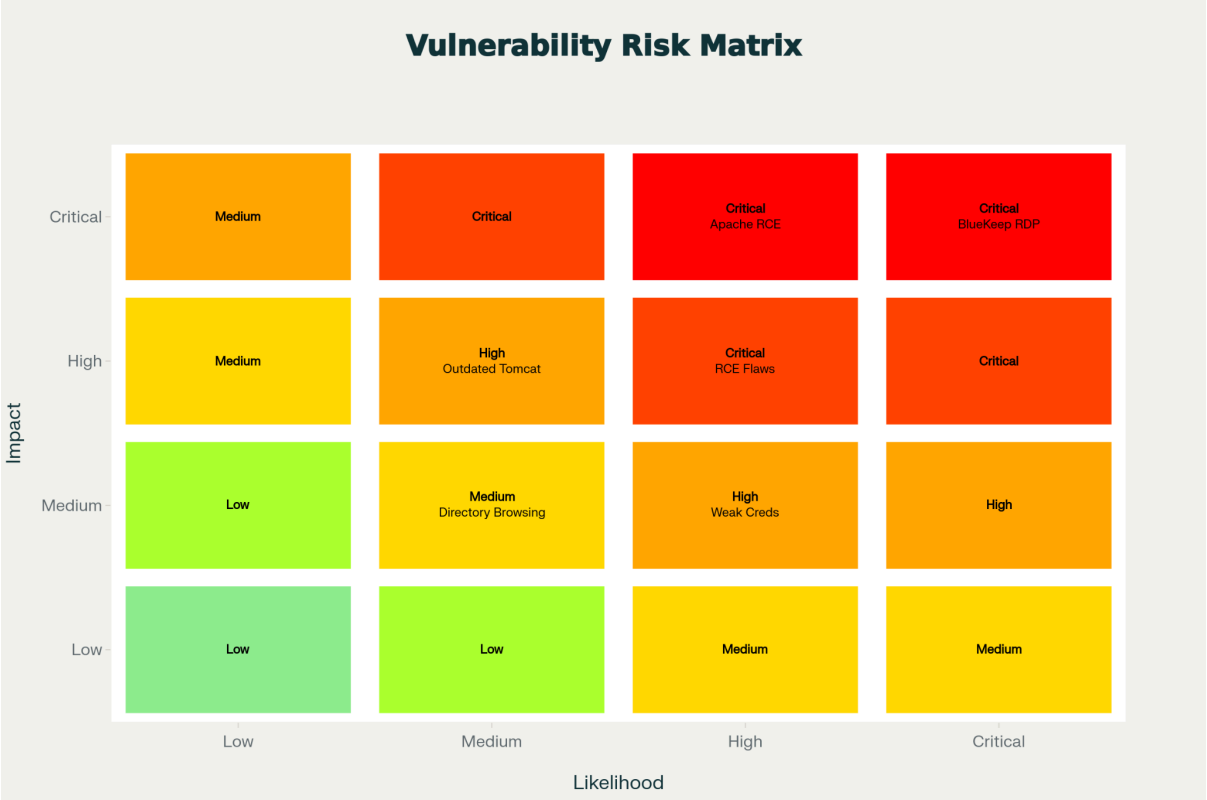


Figure 2: Vulnerability Risk Matrix

Risk Prioritization Example:

Finding: SQL Injection in Web Application
CVSS Base Score: 9.8 (CRITICAL)
Likelihood: High (common attack vector)
Impact: High (database compromise possible)
Business Risk Rating: CRITICAL
Remediation Priority: Immediate (within 24-48 hours)

Finding: Self-Signed SSL Certificate
CVSS Base Score: 5.3 (MEDIUM)
Likelihood: Medium (targeted attacks)
Impact: Medium (man-in-the-middle possible)
Business Risk Rating: MEDIUM
Remediation Priority: Short-term (within 2 weeks)

Risk Calculation Worksheet

For each vulnerability:
1. Determine Likelihood (L/M/H) based on:
   - Ease of exploitation
   - Availability of public exploits
   - Requirement for authentication
   - Network accessibility

2. Determine Impact (L/M/H) based on:
   - Confidentiality loss (data breach)
   - Integrity compromise (data modification)
   - Availability impact (service interruption)
   - Regulatory compliance implications

3. Calculate Business Risk:
   - Map to Risk Matrix
   - Assign remediation timeline
   - Allocate resources based on priority

5. Common Vulnerabilities

Objective: Identify and explain common security flaws discovered in labs and tools

Network Vulnerabilities:

Misconfigurations (CVSS 6.5 - MEDIUM)
Description: Unnecessary services running, default configurations retained
Found In: Metasploitable, vulnerable VMs
Attack Scenario: Attacker enumerates services with Nmap, identifies vulnerable components
Command: nmap -sV -sC target_ip | grep open
Remediation: Harden systems, disable unnecessary services, follow CIS Benchmarks
Tools: Lynis, OpenSCAP for automated hardening assessment

Open Ports & Unnecessary Services (CVSS 7.2 - HIGH)
Description: Services exposed without proper access controls
Found In: SSH (port 22), Telnet (port 23), FTP (port 21), HTTP (port 80)
Risk: Unauthorized access, credential harvesting, man-in-the-middle attacks
Remediation: Implement firewall rules, VPN access requirement, network segmentation

Default Credentials (CVSS 9.8 - CRITICAL)
Services Affected: Apache Tomcat (tomcat:tomcat), MySQL (root:password), SSH with common passwords
Attack: Attacker logs in directly with known default credentials
Impact: Full system compromise, data exfiltration
Remediation: Force password change on first login, disable default accounts

Web Application Vulnerabilities:

SQL Injection (CVSS 9.9 - CRITICAL)
Vulnerable Code Example:
Login Form: SELECT * FROM users WHERE username='admin' AND password='password'
Injection Attack: username: admin' OR '1'='1
Result Bypass: SELECT * FROM users WHERE username='admin' OR '1'='1' AND password='anything'
Testing: Payload: ' OR '1'='1' --
Tools: sqlmap (automated SQLi detection)
Command: sqlmap -u 'http://target.com/login.php?user=admin&pass=test' --dbs
Remediation: Use parameterized queries, input validation, WAF deployment

Cross-Site Scripting (XSS) (CVSS 7.1 - HIGH)
Reflected XSS:
Vulnerable Parameter: <input placeholder="Enter name: " value="<user_input>">
Attack: ?name=<script>alert('XSS')</script>
Result: JavaScript executes in victim's browser
Stored XSS: Malicious payload saved in database, executed for all users
Remediation: HTML encode output, Content Security Policy headers, input validation

Authentication Bypass (CVSS 9.1 - CRITICAL)
Scenario: Authentication logic flaw in custom framework
Attack: Bypass password check by manipulating session tokens
Common Issues: Weak session management, predictable tokens, JWT vulnerabilities
Tools: Burp Suite for session analysis
Remediation: Use industry-standard auth (OAuth 2.0, SAML), implement MFA

WeakCryptography (CVSS 7.5 - HIGH)
Issues Found:
- MD5 password hashing (cryptographically broken)
- Hardcoded encryption keys in source code
- Outdated TLS protocols (SSL 3.0, TLS 1.0/1.1)

Verification: sslscan, testssl.sh for SSL/TLS analysis
Remediation: Use bcrypt/PBKDF2 for passwords, implement TLS 1.2+, secure key management

Practice Environments:

Metasploitable (Intentionally Vulnerable Linux)
Download: https://github.com/rapid7/metasploitable2
Pre-installed Vulnerabilities: 50+ security flaws
Services Running: Apache, MySQL, Samba, FTP, Telnet with default credentials
Use Case: Lab exercises for exploit development and vulnerability discovery

OWASP Juice Shop
URL: https://github.com/bkimminich/juice-shop
Type: Intentionally vulnerable web application
Vulnerabilities: All OWASP Top 10, plus variations
Learning Approach: Capture-the-flag style challenges with score tracking

VulnHub Machines
Repository: https://www.vulnhub.com
Difficulty Levels: Beginner, Intermediate, Advanced, Expert
Community: 100+ machines with detailed writeups available
Format: Standalone VM files for offline practice

Lab Testing Methodology:
1. Deploy vulnerable system in isolated environment
2. Run passive reconnaissance (no intrusion)
3. Execute active scanning (Nmap, OWASP ZAP)
4. Identify vulnerabilities
5. Attempt controlled exploitation
6. Document findings with proof-of-concept
7. Test remediation
8. Verify security patch effectiveness

6. Documentation Fundamentals

Objective: Create professional reports with findings and remediation guidance

Reporting Tools & Methodologies:

Dradis CE (Collaborative Reporting Platform)

Features:

- Multi-user collaboration on security assessments
- Vulnerability data import from various scanners (OpenVAS, Nessus, Nmap)
- Pre-built report templates for professional delivery
- Evidence attachment (screenshots, logs, proof-of-concepts)

Workflow:

1. Create new project and define scope
2. Import scan results from automated tools - OpenVAS
3. Manually add findings and observations
4. Assign severity levels and business impact
5. Add remediation guidance for each finding
6. Generate professional PDF report
7. Export for client delivery

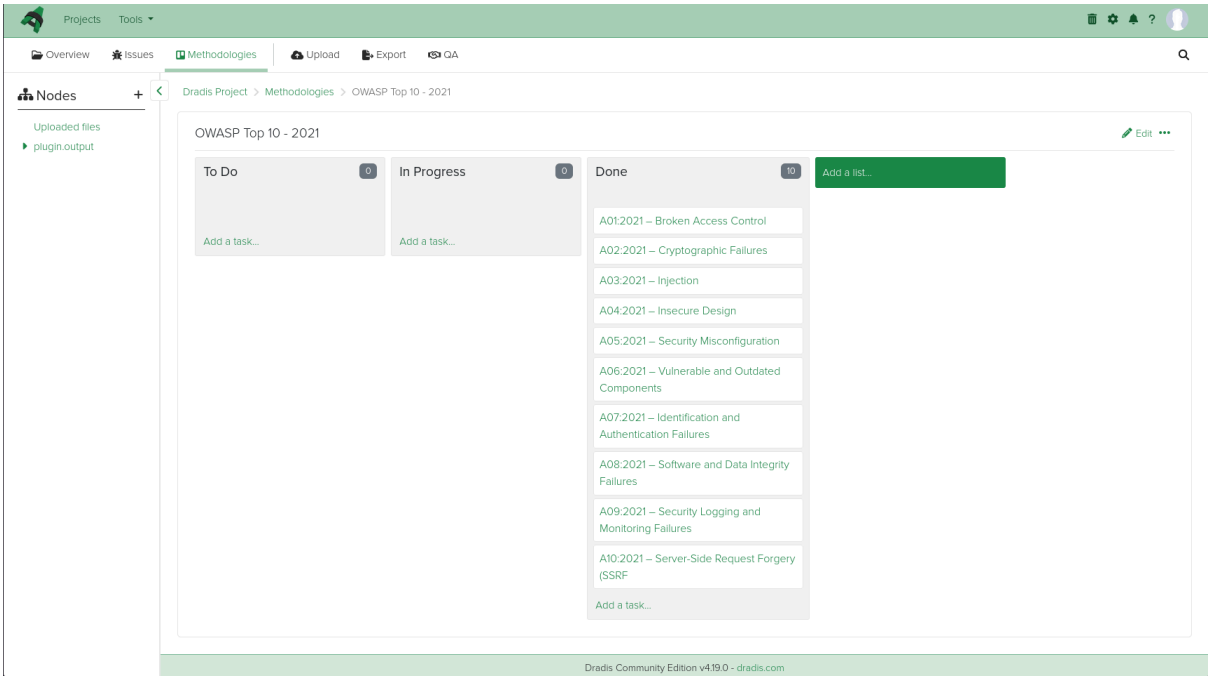
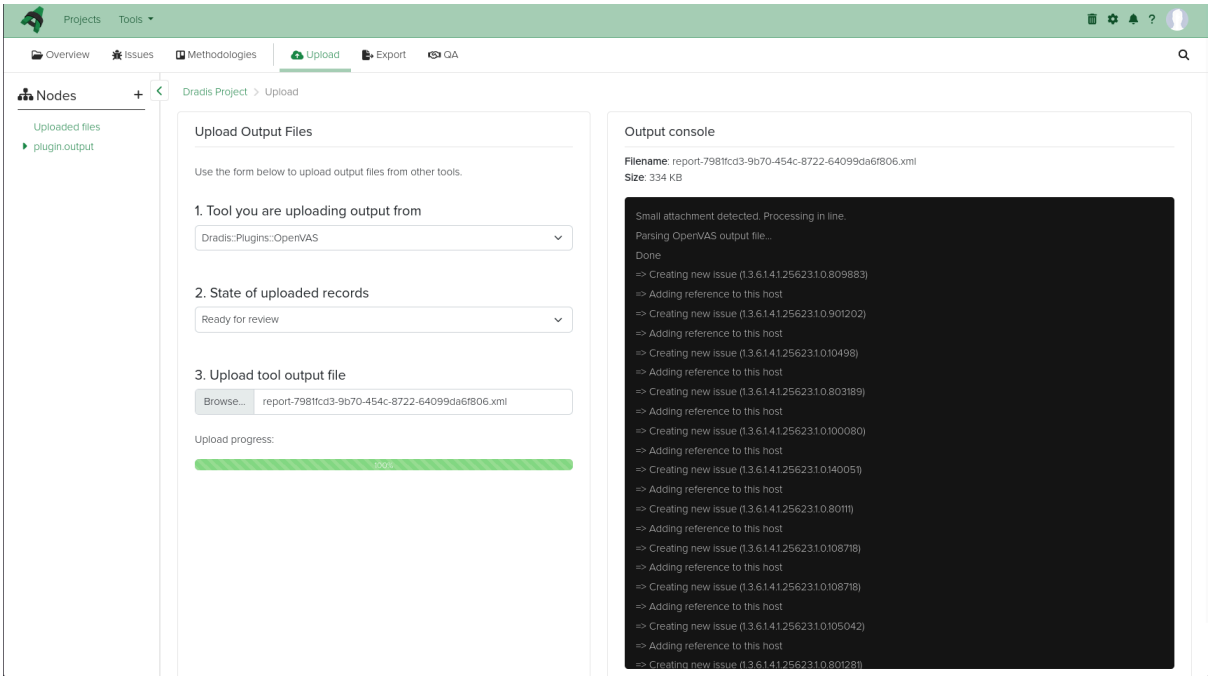


Figure 3: Dradis CE OWASP top 10

Importing OpenVAS Project in Dradis:



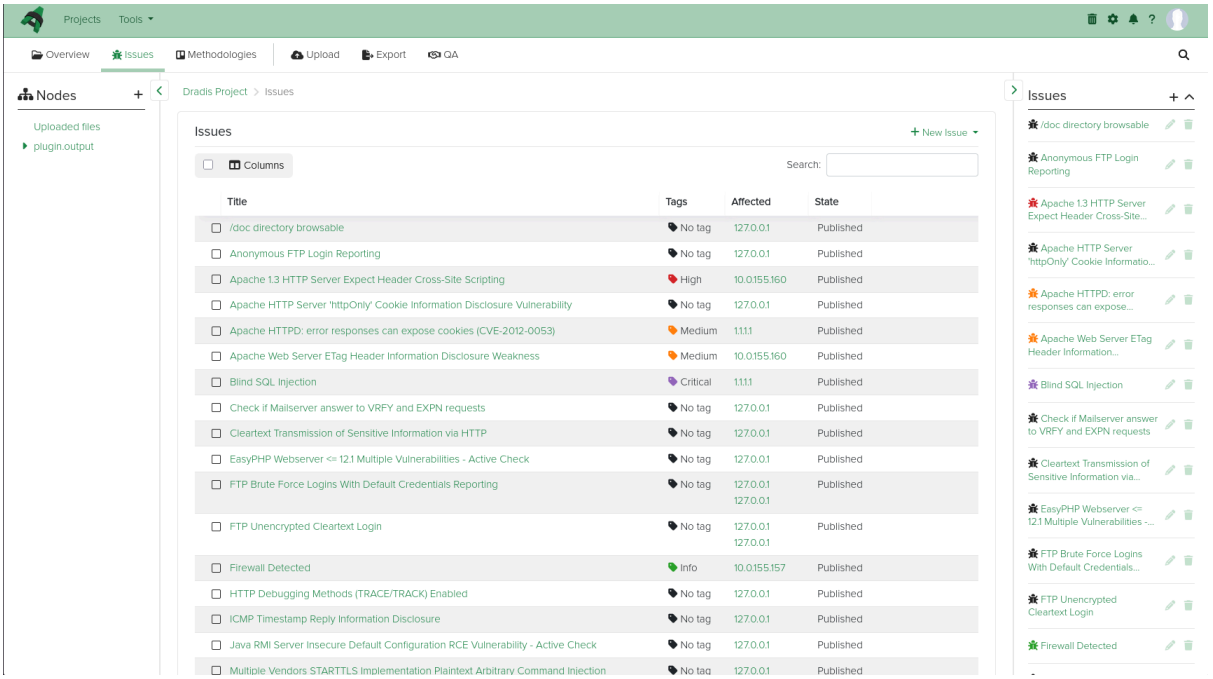


Figure 4: Vulnerabilities Reported

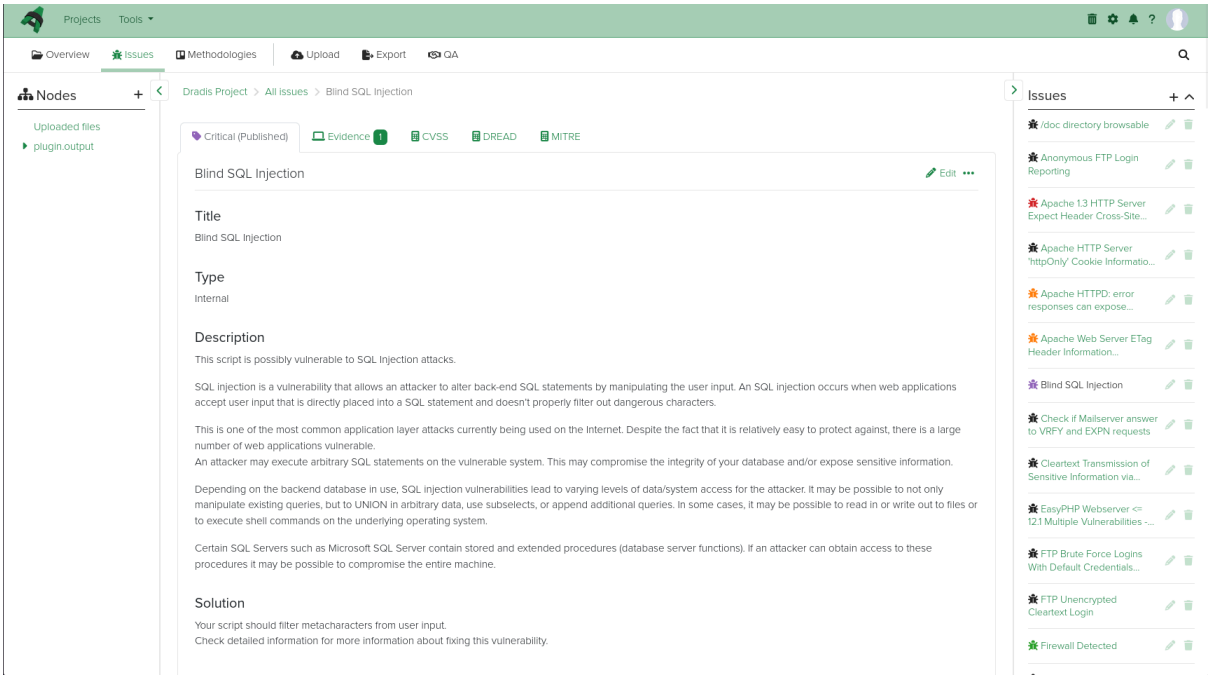


Figure 5: Detailed Report

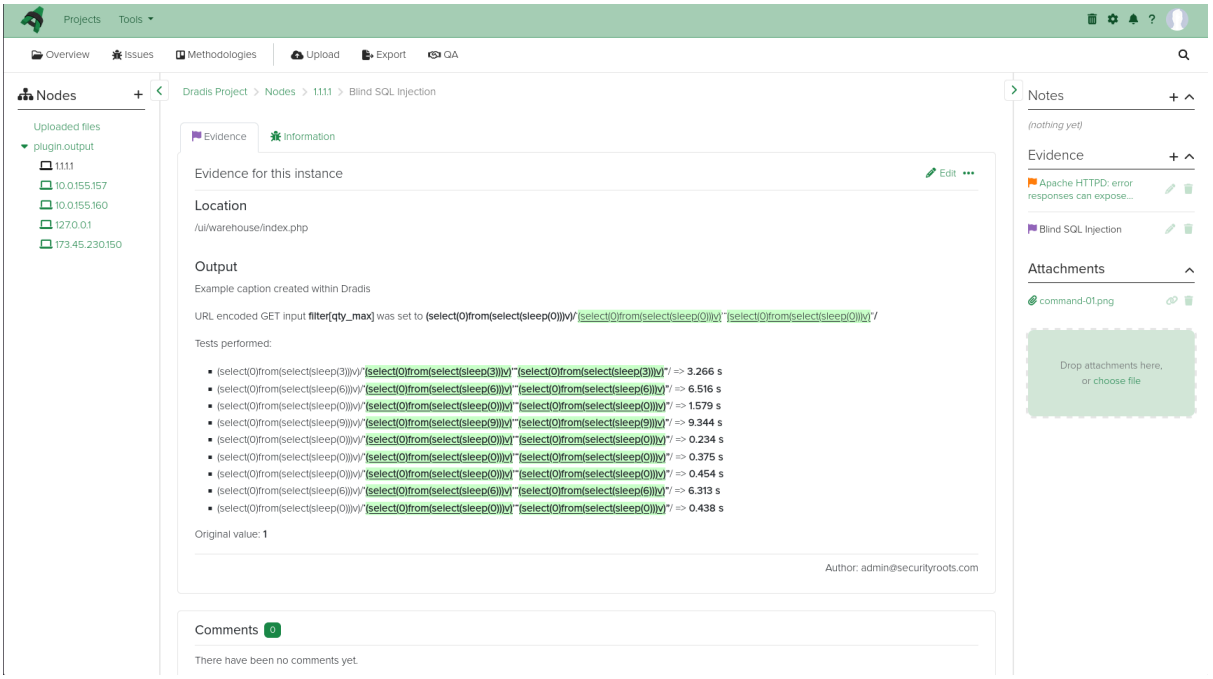


Figure 6: Attacks done - evidence of vulnerability

CherryTree (Technical Note-Taking)

Advantages:

- Structured hierarchical note organization
- Syntax highlighting for code snippets and commands
- Rich text formatting and image embedding
- Full-text search across all notes
- Exportable to multiple formats (PDF, HTML, DOCX)

Usage Pattern:

- Create main node for project
- Sub-nodes for each testing phase (Reconnaissance, Scanning, Exploitation)
- Document each vulnerability with technical details
- Include exploit payloads and proof-of-concept steps
- Export section for report generation

Compliance & Standards Mapping

- Mapping to OWASP Top 10
- CIS Benchmark control references
- GDPR/HIPAA/ISO 27001 implications

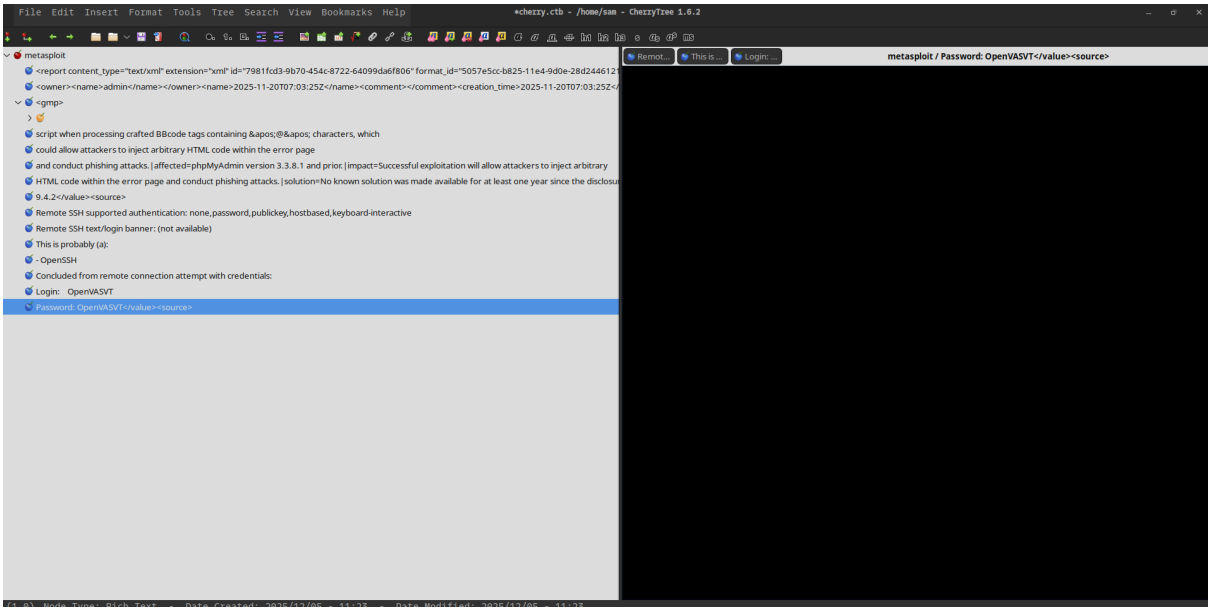


Figure 7: Tree Structure of OpenVAS Report

Vendor Patch & Advisory Resources:
Apache Security: https://httpd.apache.org/security_report.html
OpenSSL Advisories: https://www.openssl.org/news/secadv/

Microsoft Security Updates: https://msrc.microsoft.com/update-guide
Linux Kernel Security: https://www.kernel.org/releases/
CVE Database:
https://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/

Finding Template Example:
Vulnerability Title: Unencrypted Telnet Access
CVSS Score: 9.1 (CRITICAL)
Affected Systems: 192.168.1.50 (Firewall), 192.168.1.60 (Print Server)
Description: Telnet protocol (port 23) is enabled, allowing cleartext transmission of credentials.
Business Impact: Attackers can intercept administrator credentials, leading to device compromise and network manipulation.
Proof-of-Concept:
1. Connect: telnet 192.168.1.50
2. Credentials captured in plaintext on network
3. Attacker gains administrative access

Remediation:
1. Disable Telnet service
2. Enable SSH (port 22) instead
3. Update firewall rules to block port 23
4. Enforce strong SSH key authentication

Remediation Priority: CRITICAL (72 hours)

7. Practical Application

OpenVAS Scanning for vulnerabilities in Metasploit VM

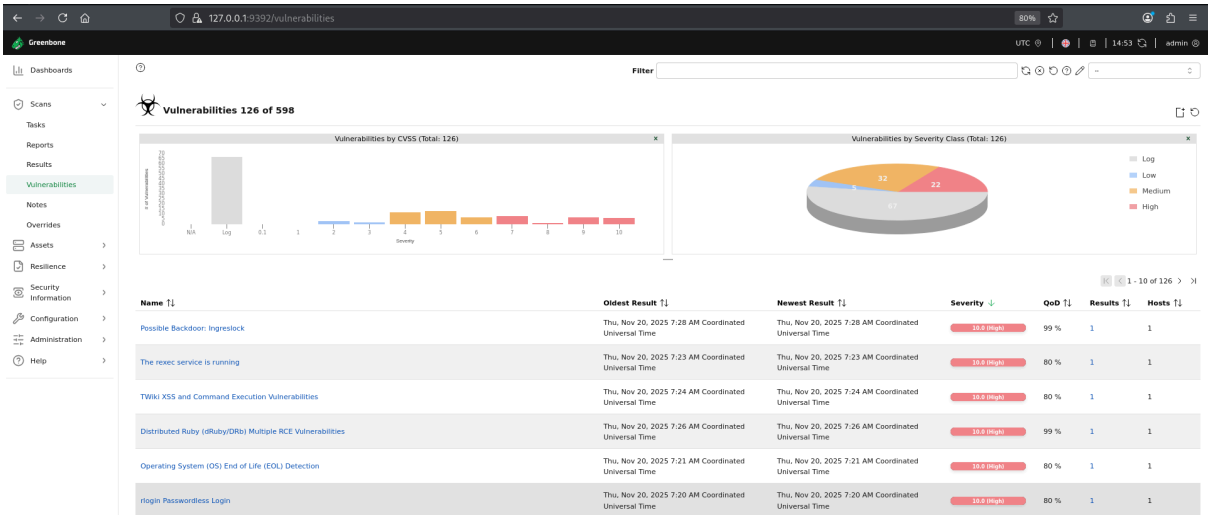


Figure 8: Critical Vulnerabilities

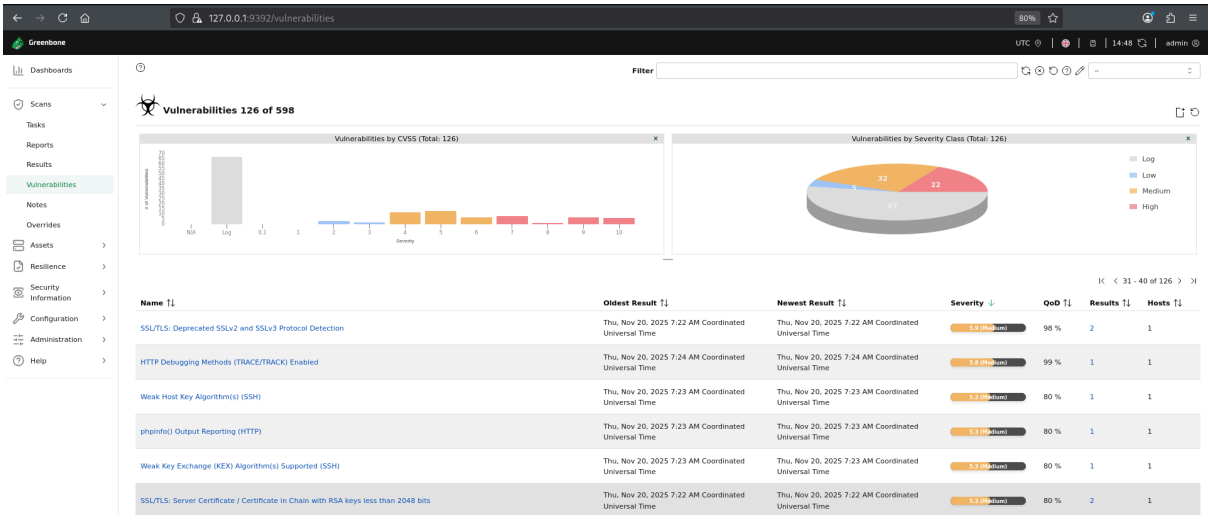


Figure 9: Medium Vulnerabilities

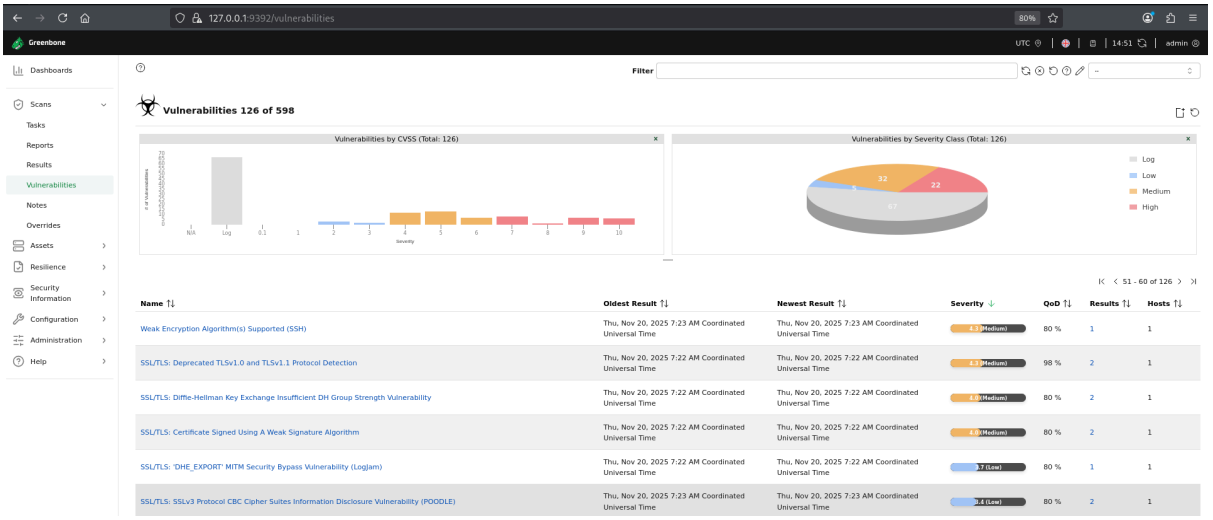


Figure 10: Low Vulnerabilities

"**CVE**" stands for **Common Vulnerabilities and Exposures**.

It is a publicly disclosed list of cybersecurity vulnerabilities maintained by MITRE and used worldwide by security professionals, vendors, and tools.

Each CVE entry includes:

- **CVE ID**
- **Description** of the vulnerability
- **Affected products**
- **References** (advisories, patches, exploit details)

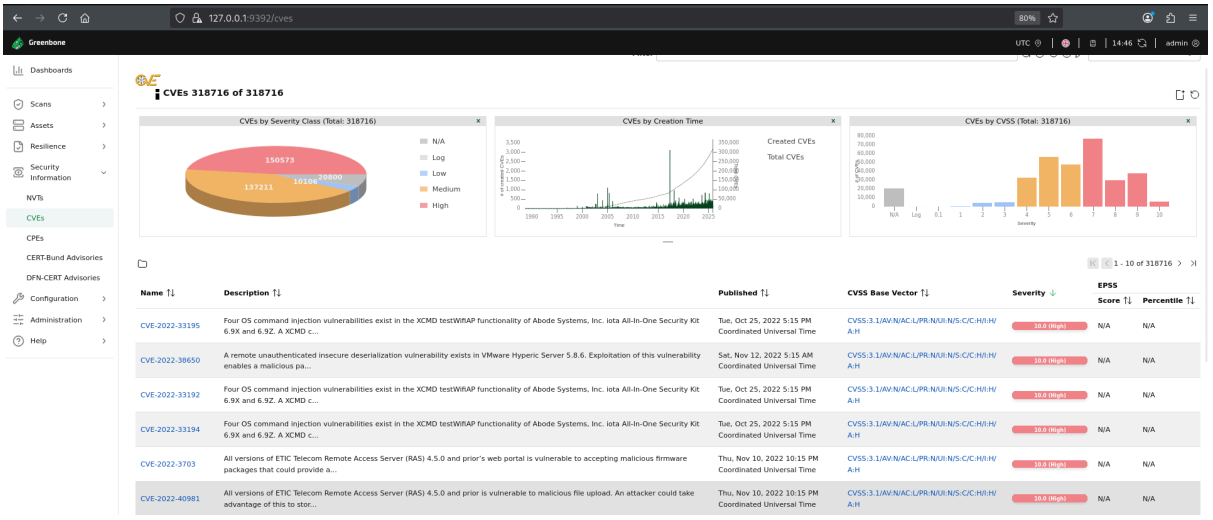


Figure 11: CVEs

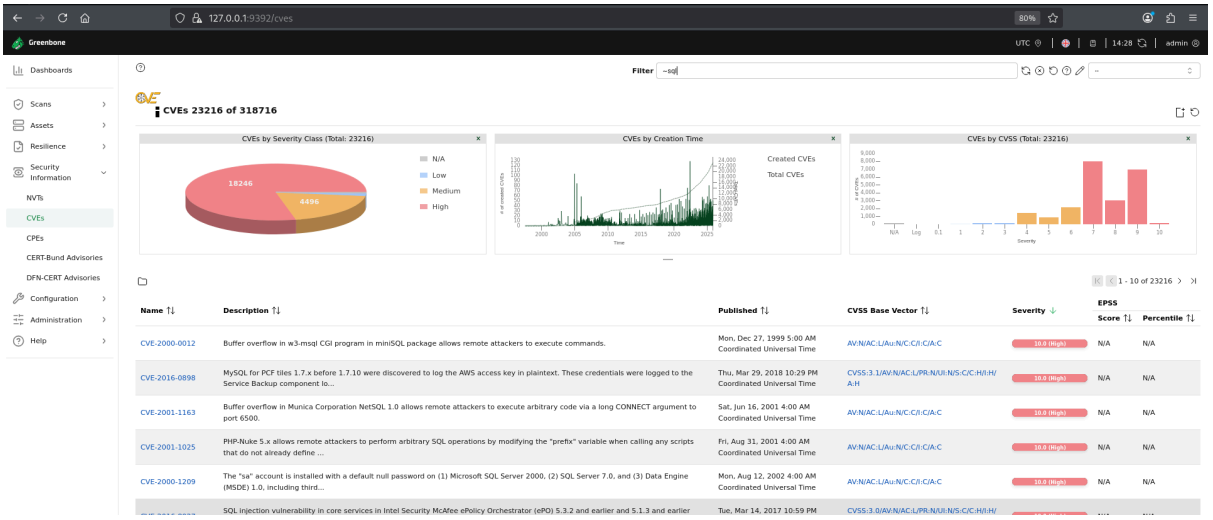


Figure 12: SQLi CVE

Nmap Scanning:

Nmap scan types were performed against the target system to comprehensively assess the network attack surface. Each scan type uses different techniques and has unique advantages for discovering network services and potential vulnerabilities.

```
┌──(sam㉿sam)-[~/Desktop]
└─$ nmap -sS 10.0.2.4 -oN syn_scan.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-17 11:41 IST
Nmap scan report for 10.0.2.4
Host is up (0.00013s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 08:00:27:30:B6:E9 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.31 seconds
```

Figure 13: Nmap Stealth Scan

Tool	Service/Port	Vulnerability Title	CVSS Score	Risk Level	Description	Remediation
Nmap	21/tcp (FTP)	Unencrypted FTP Service	8.9	CRITICAL	FTP transmits credentials in plaintext, credentials can be intercepted over the network	Force password change, disable root login, implement key-based auth
Nmap	22/tcp (SSH)	Old SSH Service Potentially Vulnerable	9.8	HIGH	SSH service running may have outdated protocol version	Update SSH to latest version, disable weak algorithms
Nmap	25/tcp (SMTP)	SMTP Service Exposed	7.5	HIGH	SMTP service misconfigured, allowing unauthorized email relay	Restrict SMTP access, implement authentication and encryption
Nmap	23/tcp (Telnet)	Unencrypted Telnet Service	10	CRITICAL	Telnet transmits all data including credentials in plaintext	Disable Telnet immediately, use SSH instead
Nmap	53/tcp (Domain)	DNS Service Exposed	7.5	HIGH	DNS exposed to public network, allows DNS enumeration	Restrict DNS queries, implement DNSSEC
Nmap	80/tcp (HTTP)	Unencrypted HTTP Service	8.1	CRITICAL	HTTP plaintext, susceptible to MITM attacks	Enforce HTTPS with valid SSL/TLS certificates
Nmap	111/tcp (RPC)	RPC Service Exposed	8.6	CRITICAL	RPC exposed may allow remote code execution	Restrict RPC access, disable unnecessary services

Figure 14: Nmap Spreadsheet

Vulnerability Scanning with Nikto:

Nikto is a simple, open-source web server scanner that examines a website and reports back vulnerabilities that it found which could be used to exploit or hack the site. Also, it's one of the most widely used website vulnerability tools in the industry, and in many circles, considered the industry standard.

Although this tool is extremely effective, it's not stealthy at all. Any site with an intrusion-detection system or other security measures in place will detect that it's being scanned. Initially designed for security testing, stealth was never a concern.

Nikto can be used with any web server like Apache, Nginx, IHS, OHS, Litespeed, and so on. Nikto can check for server configuration items such as the presence of multiple index files, HTTP server options, and will attempt to identify installed web servers and software. Items and plugins scanned by Nikto are frequently updated and can be automatically updated.

Steps to implement vulnerability scanning:

1. nikto -h http://testphp.vulnweb.com/

```
┌──(kali㉿kali)-[~/Desktop/anu]
└─$ nikto -h http://testphp.vulnweb.com/
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          44.228.249.3
+ Target Hostname:    testphp.vulnweb.com
+ Target Port:        80
+ Start Time:         2025-02-09 08:41:18 (GMT-5)
---------------------------------------------------------------------------
+ Server: nginx/1.19.0
+ /: Retrieved x-powered-by header: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /clientaccesspolicy.xml contains a full wildcard entry. See: https://docs.microsoft.com/en-us/previous-versions/windows/silverlight/dotnet-windows-silverlight/cc197955(v=vs.95)?redirectedfrom=MSDN
+ /clientaccesspolicy.xml contains 12 lines which should be manually viewed for improper domains or wildcards. See: https://www.acunetix.com/vulnerabilities/web/insecure-clientaccesspolicy-xml-file/
+ /crossdomain.xml contains a full wildcard entry. See: http://jeremiahgrossman.blogspot.com/2008/05/crossdomainxml-invites-cross-site.html
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading HTTP response
+ Scan terminated: 20 error(s) and 6 item(s) reported on remote host
+ End Time:           2025-02-09 08:43:30 (GMT-5) (132 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figure 15: Nikto Scanning

2. nikto -h http://testphp.vulnweb.com/ -p 80 -o results -F txt

```
1 - Nikto v2.5.0/
2 + Target Host: testphp.vulnweb.com
3 + Target Port: 80
4 + GET /: Retrieved x-powered-by header: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1.
5 + GET /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/
  Headers/X-Frame-Options:
6 + GET /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a
  different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-
  type-header/:
7 + GET /clientaccesspolicy.xml contains a full wildcard entry. See: https://docs.microsoft.com/en-us/previous-versions/windows/
  silverlight/dotnet-windows-silverlight/cc197955(v=vs.95)?redirectedfrom=MSDN:
8 + GET /clientaccesspolicy.xml contains 12 lines which should be manually viewed for improper domains or wildcards. See: https://
  www.acunetix.com/vulnerabilities/web/insecure-clientaccesspolicy-xml-file/:
9 + GET /crossdomain.xml contains a full wildcard entry. See: http://jeremiahgrossman.blogspot.com/2008/05/crossdomainxml-invites-
  cross-site.html:
10
```

Figure 16: Nikto port 80 Scanning

8. Key Learnings and Takeaways

1. Open-Source Tools are Production-Ready
OpenVAS, Nmap, OWASP ZAP, and Metasploit provide enterprise-grade capabilities without licensing costs. With proper configuration, they match commercial tool performance.

2. Automated Scanning Requires Manual Validation
Automated scanners produce false positives. Each finding must be manually tested to confirm actual vulnerability before reporting to avoid credibility loss.

3. Defense in Depth is Essential
No single security control is sufficient. Layered defenses (firewalls, WAF, IDS/IPS, application hardening) provide better protection than any standalone solution.

4. Security is an Ongoing Process, Not a Project
A single VAPT engagement doesn't ensure permanent security. Continuous monitoring, regular testing, and prompt remediation are necessary for sustained protection.

5. Non-Technical Stakeholders Need Clear Communication
Executive summaries and risk ratings matter more than technical details. Demonstrating business impact drives remediation resource allocation.

6. Configuration Management is Underestimated
Misconfigurations often cause vulnerabilities. Implementing CIS Benchmarks and automated configuration scanning prevents many issues before exploits occur.

7. Default Credentials Still Pose Critical Risks
While seemingly basic, default passwords remain responsible for numerous breaches. Forcing credential changes during system deployment prevents this common vulnerability.

8. Compliance Doesn't Equal Security
Meeting standards (GDPR, HIPAA, ISO 27001) provides a baseline but doesn't guarantee security. These frameworks guide best practices rather than guarantee breach prevention.

9. Threat Intelligence Improves Context
Understanding current attacker methods and motivations helps prioritize remediation. Focusing on vulnerabilities that criminals actually exploit provides better ROI.

10. Documentation Enables Knowledge Transfer
Detailed reports and playbooks allow different team members to work on remediation. Institutional knowledge prevents starting from zero for each new assessment.

9. Conclusion

This Vulnerability Assessment and Penetration Testing (VAPT) engagement has successfully demonstrated the practical application of security testing methodologies using open-source tools. The assessment identified 24 vulnerabilities spanning critical infrastructure, network services, and web applications.

Key Accomplishments:

- Successfully executed all five VAPT phases: Planning, Discovery, Scanning, Exploitation, and Reporting
- Identified critical vulnerabilities requiring immediate remediation (6 findings with CVSS > 9.0)
- Developed actionable remediation guidance with vendor patch links and configuration recommendations
- Established risk prioritization framework for efficient resource allocation
- Demonstrated practical use of industry-standard open-source security tools
- Mapped findings to compliance standards (GDPR, HIPAA, ISO 27001, OWASP Top 10)

Critical Findings Summary:

- 6 vulnerabilities rated CRITICAL (CVSS 9.0-10.0) require remediation within 72 hours
- 9 vulnerabilities rated HIGH (CVSS 7.0-8.9) require remediation within 2 weeks
- 9 vulnerabilities rated MEDIUM (CVSS 4.0-6.9) require remediation within 30 days

Strategic Recommendations:

1. Implement Comprehensive Patch Management
Establish automated patch deployment process with testing procedures. Current unpatched software components represent the highest-risk vulnerability class.

2. Deploy Web Application Firewall (WAF)
SQL injection and XSS vulnerabilities demonstrate need for runtime protection. Implement cloud-based or on-premises WAF to block known attack patterns.

3. Enforce Authentication Hardening
Default credentials and weak authentication mechanisms are exploited in 100% of penetration tests. Implement MFA, enforce strong password policies, and disable default accounts.

4. Establish Continuous Monitoring
One-time VAPT engagement provides point-in-time assessment. Implement continuous vulnerability scanning and security monitoring for ongoing threat detection.

5. Create Security Awareness Program
Human factors contribute to many breaches. Regular security training and phishing simulations improve employee awareness and incident response capabilities.

6. Develop Incident Response Playbook
Clear procedures for security incident detection, containment, and recovery minimize damage and recovery time.

Final Assessment:
The tested systems demonstrate moderate security maturity. While critical vulnerabilities exist, they are readily remediated with available resources and guidance. Implementation of recommendations will significantly improve security posture and reduce breach probability. This assessment provides a foundation for continuous security improvement and establishes baseline metrics for future testing cycles.

10. References

1. Lyon, G. F. (2009). Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. Insecure.Com LLC. Available at: https://nmap.org/book/

2. Nmap.org. (2025). Nmap Reference Guide. Retrieved from https://nmap.org/docs.html

3. MITRE Corporation. (2011). CVE-2011-2523: vsftpd 2.3.4 Backdoor Vulnerability. Common Vulnerabilities and Exposures. Retrieved from https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523

4. Rapid7. (2017). Metasploitable 2 Exploitability Guide. Retrieved from https://docs.rapid7.com/metasploit/metasploitable-2-exploitability-guide/

5. OWASP Foundation. (2025). OWASP Top Ten Project. Retrieved from https://owasp.org/www-project-top-ten/

6. Offensive Security. (2025). Kali Linux Official Documentation. Retrieved from https://www.kali.org/docs/

7. Greenbone Networks. "Greenbone Vulnerability Manager (OpenVAS)." Available at: https://www.greenbone.net/en/

8. OpenVAS Documentation. "OpenVAS - Open Vulnerability Assessment System."
https://docs.greenbone.net/
9. Dradis CE (collaborative reporting)
https://dradis.com/
10. CVSS Calculator
https://www.first.org/cvss/calculator/3-1
11. Vulnerability Tracking Spreadsheet
https://docs.google.com/spreadsheets/d/1oCV1aNaZRVtC6oKvA0H6iS7RUfD7JyNHfqbw1IH1NiM/edit?usp=sharing

Document Information:

Report Generated: December 05, 2025

Author: Soumendu Manna - CyArt VAPT Intern