Web Application Vulnerability Assessment Report

Target: Metasploitable2 Test Environment
Assessment Date: November 28, 2025
Assessor: Dhruv Shelar
Scope: Web Application Security Assessment
Status: CONFIDENTIAL - For Educational Purposes Only

Executive Summary

A comprehensive web application vulnerability assessment was conducted on the Metasploitable2 test environment. The assessment identified multiple critical and high-severity vulnerabilities that pose significant security risks. This report documents all findings, their potential impact, and recommended remediation strategies.

Key Findings:

- 1 Critical Vulnerability
- 3 High-Severity Vulnerabilities
- 2 Medium-Severity Vulnerabilities
- Multiple directory disclosures and misconfigurations

Overall Risk Rating: CRITICAL

---

1. Assessment Methodology

1.1 Reconnaissance & Discovery

- Web server fingerprinting and banner grabbing
- Directory enumeration and path discovery
- HTTP method analysis
- Web application mapping

1.2 Scanning & Enumeration

- Automated vulnerability scanning using industry-standard tools
- Manual testing and validation of findings
- Protocol analysis and security header inspection
- Service version identification

1.3 Analysis & Reporting

- Vulnerability classification using CVSS scoring
- Impact assessment and exploitation feasibility
- Risk prioritization based on severity and exploitability
- Remediation recommendations

---

2. System Identification

Target IP Address: 10.10.X.X (Lab-internal)
Port: 80 (HTTP)
Web Server: Apache httpd 2.2.8 (Ubuntu)
Web Server Modules: DAV/2, mod_proxy_ftp
Backend Technology: PHP 5.2.4-2ubuntu5.10
Operating System: Ubuntu Linux

---

3. Vulnerability Findings

3.1 Critical Severity Vulnerabilities

3.1.1 Cross-Site Tracing (XST) Vulnerability

Vulnerability ID: XST-001
CVSS Score: 7.5 (High)
Status: CONFIRMED

Description:
The target web server is vulnerable to Cross-Site Tracing (XST) attacks. The HTTP TRACE method is enabled on the server, which allows attackers to bypass HTTPOnly cookie restrictions and access sensitive information transmitted via HTTP requests.

Technical Details:

- HTTP TRACE method is not disabled
- Server echoes request headers back to the client
- Can be exploited to steal session cookies and authentication tokens
- Potential vector for session hijacking attacks

Attack Vector:
An attacker can craft a malicious HTML page containing JavaScript that sends TRACE requests to the vulnerable server. The server's response would include headers and cookies, allowing the attacker to extract sensitive session information.

Potential Impact:

- **Confidentiality:** HIGH - Session cookies and authentication tokens can be stolen
- **Integrity:** MEDIUM - Potential for unauthorized modifications using stolen credentials
- **Availability:** MEDIUM - Can be used for Denial of Service attacks
- **Business Impact:** User account compromise, unauthorized transactions, data breach

Exploitation Difficulty: EASY - Can be exploited using basic AJAX requests

Proof of Concept:
curl -i -X TRACE http://10.10.X.X:80/
Expected Response: Request headers echoed back with session information

Remediation:

1. Disable the HTTP TRACE method in Apache configuration:
    - Add to httpd.conf: `TraceEnable Off`
2. Implement strict HTTP security headers
3. Use HttpOnly flag on all session cookies
4. Enable HTTPS/TLS for all communications
5. Implement Content Security Policy (CSP)

Priority: CRITICAL - Must be remediated immediately

3.1.2 Outdated Apache Web Server (2.2.8)

Vulnerability ID: SYS-001
CVSS Score: 8.1 (Critical)
Status: CONFIRMED

Description:
The target system is running Apache httpd version 2.2.8, released in 2008. This version contains multiple known vulnerabilities, including XSS flaws, directory traversal issues, and denial of service vulnerabilities. Apache 2.2 reached End-of-Life in December 2017 and receives no security updates.

Known CVEs in Apache 2.2.8:

- mod_proxy_ftp UTF-7 XSS (CVE-2008-0005)
- mod_status XSS vulnerability (affecting status pages)
- mod_imagemap XSS vulnerability (CVE-2007-5000)
- Multiple denial of service vulnerabilities

Potential Impact:

- Remote code execution (RCE) through crafted requests
- Arbitrary file disclosure
- Cross-site scripting attacks
- Server crash and denial of service
- Complete system compromise

Remediation:

1. Upgrade to Apache 2.4.x (latest stable version)
    - Apache 2.4 receives ongoing security patches
    - Improved performance and security features
2. If immediate upgrade impossible: Apply WAF rules to block known exploit patterns
3. Implement strict access controls to limit exposure
4. Disable unnecessary Apache modules (mod_status, mod_proxy_ftp)

Priority: CRITICAL - Affects entire web server infrastructure

3.2 High-Severity Vulnerabilities

3.2.1 Outdated PHP Version (5.2.4)

Vulnerability ID: APP-001
CVSS Score: 7.8 (High)
Status: CONFIRMED

Description:
The target application uses PHP version 5.2.4-2ubuntu5.10, released in 2007. This version is extremely outdated and contains numerous critical vulnerabilities including remote code execution flaws, SQL injection issues, and insecure cryptographic functions.

Affected Components:

- Session handling (potential session fixation)
- File operations (potential LFI/RFI)
- Cryptographic functions (weak or deprecated)
- Database interactions (vulnerable to injection)

Known Vulnerabilities in PHP 5.2.4:

- Multiple buffer overflow vulnerabilities
- SQL injection in standard functions
- Information disclosure through error messages
- Insecure deserialization

Potential Impact:

- Remote code execution on the server
- Complete application compromise
- Database access and manipulation
- Sensitive information disclosure
- Server-side template injection attacks

Remediation:

1. Upgrade to PHP 8.x or minimum PHP 7.4 (which receives security patches)
2. Review and update all application code for compatibility
3. Enable PHP security features:
    - disable_functions directive to restrict dangerous functions
    - open_basedir restriction
    - expose_php = Off
4. Implement input validation and output encoding
5. Use parameterized queries (prepared statements)

Priority: CRITICAL - Active web application framework

3.2.2 WebDAV (Distributed Authoring and Versioning) Information Disclosure

Vulnerability ID: DAV-001
CVSS Score: 6.5 (Medium-High)
Status: CONFIRMED

Description:
WebDAV is enabled on the target server with directory listing enabled. The following directories are accessible and potentially exploitable:

Accessible WebDAV Directories:

- http://10.10.X.X:80/dav/ (HTTP 200 - Accessible)
- Directory enumeration possible via PROPFIND requests

Attack Vectors:

1. **Directory Listing Enumeration:** Attackers can use PROPFIND method to list directory contents
2. **Unauthorized File Access:** Potentially read files in WebDAV-enabled directories
3. **File Upload:** Depending on permissions, attackers may upload malicious files
4. **File Modification:** WebDAV allows PUT and DELETE operations if permissions allow

Exploitation Techniques:
Using curl to enumerate WebDAV directories:
curl -X PROPFIND http://10.10.X.X:80/dav/

Potential Impact:

- Information disclosure about directory structure
- Potential file manipulation or upload of malicious content
- Unauthorized access to sensitive files
- Lateral movement within file system

Remediation:

1. Disable WebDAV if not required:
    - Comment out DAV modules in httpd.conf
    - Disable mod_dav and mod_dav_fs
2. If WebDAV is required:
    - Restrict access using Apache authentication
    - Implement strict file permissions
    - Use HTTPS/TLS for all WebDAV connections
    - Disable directory listing
    - Implement access control lists (ACLs)
3. Monitor WebDAV operations for suspicious activity

Priority: HIGH - Potential unauthorized file access

3.2.3 phpMyAdmin Directory Discovery

Vulnerability ID: APP-002
CVSS Score: 7.2 (High)
Status: CONFIRMED

Description:
The phpMyAdmin administration interface is publicly accessible and discoverable at:

- http://10.10.X.X:80/phpMyAdmin/ (HTTP 301 Redirect)
- http://10.10.X.X:80/phpMyAdmin (HTTP 301 Redirect)

phpMyAdmin is a web-based MySQL administration tool. Exposure of this interface directly to the internet is a critical security misconfiguration.

Attack Scenarios:

1. **Brute Force Attack:** Attackers can attempt to brute-force database credentials
2. **SQL Injection:** phpMyAdmin versions may contain SQL injection vulnerabilities
3. **Information Disclosure:** Database structure and sensitive information exposure
4. **Unauthorized Database Access:** Complete database compromise possible

Potential Impact:

- Complete database access and manipulation
- Extraction of sensitive customer/business data
- Unauthorized record modifications
- Database deletion or corruption
- Compliance violations (GDPR, PCI-DSS, HIPAA)

Risk Assessment:

- **Likelihood:** HIGH - Known target for automated scanning
- **Impact:** CRITICAL - Complete data compromise possible

Remediation:

1. Remove phpMyAdmin from internet-accessible location
2. If administration interface required:
    - Move to restricted IP address accessible only from admin networks
    - Implement strong authentication (multi-factor if possible)
    - Use VPN or IP whitelisting
    - Rename the directory to non-standard name
    - Implement Web Application Firewall (WAF) rules
3. Use alternative administration methods (SSH tunneling, secure VPN)
4. Disable direct access:
   <Directory /var/www/html/phpMyAdmin>
   Deny from all
   </Directory>

Priority: CRITICAL - Database administration interface exposure

---

3.3 Medium-Severity Vulnerabilities

3.3.1 Directory Enumeration and Path Disclosure

Vulnerability ID: CONFIG-001
CVSS Score: 5.3 (Medium)
Status: CONFIRMED

Description:
Multiple directories on the target server are accessible and discoverable through HTTP requests:

Discovered Directories:

Directory	HTTP Status	Assessment
/dav/	200 OK	Accessible - WebDAV directory
/doc/	200 OK	Accessible - Documentation
/icons/	200 OK	Accessible - Icon files
/index/	200 OK	Accessible - Index directory
/cgi-bin/	403 Forbidden	Protected but discoverable

Enumeration Results:

- Web root structure is partially mapped
- Directory listing appears to be enabled in some locations
- Application paths are revealed to attackers

Attack Implications:

- Attackers gain insight into application structure
- Easier to identify attack surface
- May reveal backup or hidden configuration files
- Information aids in targeted exploitation

Remediation:

1. Disable directory listing:
   <Directory /var/www/html>
   Options -Indexes
   </Directory>
2. Configure error handling to prevent path disclosure:
   ErrorDocument 403 "Access Denied"
   ErrorDocument 404 "Not Found"
3. Remove unnecessary directories from web root
4. Use robots.txt to guide crawlers (secondary measure only)
5. Implement index files in all accessible directories

Priority: MEDIUM - Reconnaissance aid for attackers

3.3.2 Missing Security Headers

Vulnerability ID: CONFIG-002
CVSS Score: 5.8 (Medium)
Status: CONFIRMED

Description:
The target web server is missing critical HTTP security headers. Analysis of HTTP responses indicates absence of:

Missing Security Headers:

- X-Frame-Options (Clickjacking protection)
- X-Content-Type-Options (MIME type sniffing protection)
- Strict-Transport-Security (HSTS)
- Content-Security-Policy (CSP)
- X-XSS-Protection
- Referrer-Policy

Impact:

- Increased vulnerability to clickjacking attacks
- MIME type confusion attacks possible
- Client-side code execution risks
- Session fixation risks

Remediation:
Add the following headers to Apache configuration:
Header set X-Frame-Options "SAMEORIGIN"
Header set X-Content-Type-Options "nosniff"
Header set X-XSS-Protection "1; mode=block"
Header set Referrer-Policy "strict-origin-when-cross-origin"
Header set Content-Security-Policy "default-src 'self'"
Header set Strict-Transport-Security "max-age=31536000; includeSubDomains"

Priority: MEDIUM - Defense-in-depth improvement

---

4. Vulnerability Summary Table

ID	Vulnerability	Severity	CVSS	Status	Affected Component
XST-001	Cross-Site Tracing	CRITICAL	7.5	CONFIRMED	Apache httpd
SYS-001	Outdated Apache 2.2.8	CRITICAL	8.1	CONFIRMED	Web Server
APP-001	Outdated PHP 5.2.4	CRITICAL	7.8	CONFIRMED	Application
APP-002	phpMyAdmin Exposure	CRITICAL	7.2	CONFIRMED	Application
DAV-001	WebDAV Information Disclosure	HIGH	6.5	CONFIRMED	Apache DAV
CONFIG-001	Directory Enumeration	MEDIUM	5.3	CONFIRMED	Web Server
CONFIG-002	Missing Security Headers	MEDIUM	5.8	CONFIRMED	Web Server

---

5. Risk Assessment Matrix

SEVERITY vs EXPLOITABILITY

```
┌──────────────────────────────────────────────────────────────────────┐
│ CRITICAL (Exploit Immediately) │
│ - XST (Easy to exploit) │
│ - Outdated Apache (Known exploits available) │
│ - Outdated PHP (Public RCE exploits) │
│ - phpMyAdmin Exposure (High profile target) │
├──────────────────────────────────────────────────────────────────────┤
│ HIGH (Exploit Soon) │
│ - WebDAV Information Disclosure (Medium effort) │
├──────────────────────────────────────────────────────────────────────┤
│ MEDIUM (Exploit Eventually) │
│ - Directory Enumeration (Reconnaissance aid) │
│ - Missing Security Headers (Defense bypass) │
└──────────────────────────────────────────────────────────────────────┘
```

---

6. Remediation Roadmap

Phase 1: Immediate Actions (0-7 days)

1. Disable HTTP TRACE method (XST mitigation)
2. Disable phpMyAdmin public access
3. Disable WebDAV if not needed
4. Add security headers to responses
5. Document all findings and impacts

Phase 2: Short-term Actions (1-4 weeks)

1. Plan and execute Apache upgrade to 2.4.x
2. Plan and execute PHP upgrade to 7.4 or 8.x
3. Implement WAF (Web Application Firewall)
4. Configure strict access controls
5. Enable HTTPS/TLS for all communications

Phase 3: Long-term Actions (1-3 months)

1. Application code audit and remediation
2. Security testing and validation
3. Implement security monitoring
4. Establish patch management process
5. Regular vulnerability assessments

---

7. Security Recommendations

7.1 Immediate Security Hardening

1. **Update all software** to latest stable versions with security patches
2. **Enable HTTPS/TLS** for all connections
3. **Implement authentication** for administrative interfaces
4. **Disable unnecessary services** and modules
5. **Enable security logging** and monitoring

7.2 Access Control

1. Implement principle of least privilege
2. Use IP whitelisting for administrative access
3. Implement multi-factor authentication where applicable
4. Regular access review and audit

7.3 Monitoring and Detection

1. Implement Security Information and Event Management (SIEM)
2. Enable detailed access logging
3. Monitor for suspicious HTTP methods (TRACE, DELETE, PUT)
4. Alert on failed authentication attempts
5. Regular log review and analysis

7.4 Defense in Depth

1. Implement Web Application Firewall (WAF)
2. Use intrusion detection systems (IDS)
3. Regular vulnerability scanning
4. Penetration testing
5. Security awareness training

---

8. Compliance Implications

Standards Affected:

- OWASP Top 10 (A01:2021 Broken Access Control, A06:2021 Vulnerable Components)
- CWE-1035 (Vulnerable 3rd Party Component)
- PCI-DSS Requirement 6.2 (Security patches)
- NIST Cybersecurity Framework

Potential Compliance Issues:

- Regular patching requirement violations
- Security control gaps
- Audit trail inadequacy
- Unencrypted sensitive data transmission

---

9. Conclusion

The Metasploitable2 test environment contains multiple critical vulnerabilities that require immediate remediation. The primary issues stem from the use of outdated, unsupported software versions (Apache 2.2.8, PHP 5.2.4) and misconfigurations (exposed phpMyAdmin, enabled XST, active WebDAV).

Critical Actions Required:

1. Immediate patching and updates
2. Remove public exposure of administrative interfaces
3. Disable unnecessary services
4. Implement security controls
5. Enable monitoring and logging

Timeline: All critical vulnerabilities should be remediated within 7 days to reduce security risk.

Assessment Conclusion: The target environment demonstrates significant security weaknesses typical of legacy/test systems. Proper security practices including regular updates, access controls, and defense-in-depth strategies would substantially reduce attack surface and risk exposure.

---

10. Appendix

A. Tools and Methodology

- Directory enumeration via HTTP scanning
- Web server fingerprinting and banner analysis
- HTTP method testing
- Known vulnerability database cross-reference
- Manual exploitation testing

B. Testing Standards

- OWASP Testing Guide v4.2
- NIST SP 800-115 (Technical Security Testing)
- CEH Methodology (Scanning, Enumeration, Exploitation)
- Industry best practices for vulnerability assessment

Report Generated: November 28, 2025
Report Status: CONFIDENTIAL - Educational/Lab Testing Only
Distribution: Authorized Personnel Only

This assessment was conducted on an intentionally vulnerable test environment (Metasploitable2) for educational and training purposes. All findings are specific to this lab environment and the techniques used demonstrate common web security vulnerabilities and attack methodologies.