Vulnerability Assessment Report Using OWASP ZAP

Task 3 – Web Application "Mystery Box" Assessment

Submitted by API AKSHAYA A

---

1. Introduction

This assessment was performed on a vulnerable test environment using **Metasploitable2** as the target web application and **OWASP ZAP** as the security scanning tool. The objective of this task is to identify common web application vulnerabilities and understand their risks using an automated vulnerability scanner.

OWASP ZAP (Zed Attack Proxy) is an open-source web application security tool that helps detect weaknesses such as injection flaws, security misconfigurations, missing headers, and sensitive information disclosures.

---

2. Tools & Environment

Component	Description
Attacker Machine	Kali Linux (VirtualBox)
Target Web Server	Metasploitable2
Scanning Tool	OWASP ZAP
Target URL	http://192.168.56.101

Both virtual machines were configured using Host-Only networking to allow direct communication, while the Kali VM also used NAT mode for internet connectivity.

3. Methodology

The assessment followed these steps:

1. Started the Metasploitable2 VM and confirmed the web services were accessible.
2. Launched OWASP ZAP on Kali Linux.
3. Used the **Automated Scan** feature and provided the target URL.
4. ZAP crawled all reachable pages, analyzed parameters, tested inputs, and scanned for common vulnerabilities.
5. Identified issues were listed under the **Alerts** tab and categorized based on severity.

This approach helps quickly identify “low-hanging fruit” vulnerabilities without manual penetration testing.

---

4. Scan Results Overview

OWASP ZAP reported **22 alerts** across various severity levels.

Severity	Count
🔴 **High**	Few
🟠 **Medium**	Several

Severity	Count
🟡 **Low**	Many
🔵 **Informational**	Many

The vulnerabilities discovered include a mix of security misconfigurations, missing security headers, information disclosures, and weak cryptographic practices.

---

5. Key Vulnerabilities & Risk Explanation

🔶 1. Hash Disclosure – MD5 Crypt (Medium)

Description:
The server exposed hashed password strings using MD5.
Risk:
MD5 is a weak hashing algorithm and can be cracked, leading to credential compromise.

🔶 2. Absence of Anti-CSRF Tokens (Medium)

Description:
Forms or actions on the site do not contain CSRF protection tokens.
Risk:
An attacker can trick authenticated users into performing unwanted actions (CSRF attack).

🔶 3. Application Error Disclosure (Medium)

Description:
The application reveals detailed error messages.
Risk:
Attackers can learn about server paths, frameworks, and internal logic.

🔶 4. Content Security Policy (CSP) Header Not Set (Medium)

Description:
The server does not enforce CSP.
Risk:
Significantly increases vulnerability to XSS attacks.

🔶 5. Directory Browsing Enabled (Low)

Description:
Folders can be viewed directly in the browser.
Risk:
Sensitive files or configurations may be exposed.

🔶 6. Missing Anti-Clickjacking Header (Low)

Description:
No X-Frame-Options or CSP headers are present.
Risk:
Allows clickjacking attacks, tricking users into clicking hidden UI elements.

🔶 7. Vulnerable JavaScript Library (Medium)

Description:
The website uses outdated JS files with known vulnerabilities.
Risk:
May enable DOM-based attacks or XSS.

🔶 8. Cookie Without HttpOnly Flag (Low)

Description:
Cookies can be accessed from JavaScript.
Risk:
Stolen cookies → session hijacking.

🔶 9. Cookie Without SameSite Attribute (Low)

Description:
Cookies are sent in cross-site requests.
Risk:
Can be abused in CSRF attacks.

🔶 10. Information Disclosure – Debug Messages (Medium)

Description:
Debug or stack trace information shown in pages.

Risk:
Helps attackers craft targeted exploits.

🔶 11. Private IP Disclosure (Informational)

Description:
Server leak internal IP addresses.
Risk:
Reveals network structure to attackers.

🔶 12. Server Leaks Information via HTTP Response Headers (Low)

Description:
Headers like “Server” and “X-Powered-By” expose version details.
Risk:
Attackers can match versions with known CVEs.

🔶 13. Timestamp Disclosure – Unix (Informational)

Description:
Server displays timestamps.
Risk:
Can assist in fingerprinting / analyzing server behavior.

🔶 14. X-Content-Type-Options Header Missing (Low)

Description:
Server does not prevent MIME-type sniffing.
Risk:
Can enable malicious file interpretation.

🔶 15. Sensitive Information in URL (Medium)

Description:
Parameters include sensitive or internal information.
Risk:
URLs may leak data through logs, history, or referrers.

🔶 16–22. Additional Information Disclosure Alerts

These include suspicious comments, internal references, and misconfigured security headers that collectively increase attack surface.

---

6. Conclusion

The OWASP ZAP scan revealed multiple vulnerabilities, many of which are due to outdated software, insecure default configurations, and missing security headers. These findings demonstrate the importance of:

- Implementing secure coding practices
- Disabling directory browsing
- Using strong cryptographic protections

- Sanitizing inputs and error messages
- Configuring security headers properly

This exercise provides valuable hands-on experience in using automated scanning tools and understanding real-world web application risks.