Secure Software Design Project Week-7 Deliverable

Saad Ali 2022512
Arsalan Khan 2022115
Cyber Security 32

Submitted to: Dr Zubair Ahmad

Introduction

This report presents the findings of Week 7’s Security Testing & Vulnerability Analysis for the ride-hailing application. Using OWASP ZAP and manual penetration testing, we evaluated the effectiveness of recently implemented security controls—including password complexity checks, session timeouts, security headers, and rate limiting—while identifying residual risks. The analysis compares results from initial and follow-up scans, highlighting mitigated vulnerabilities and prioritizing actions for Week 8.

1. Security Improvements Implemented

Feature	Test Method	Result	ZAP Alert Impact
Password Complexity	Manual input testing	Blocks weak passwords (<8 chars, no special chars)	N/A (Logical control)
Session Timeout	Session hijacking test	Auto-logout after 30 mins	Reduced session fixation risks
Security Headers	ZAP passive scan	X-Frame-Options, X-Content-Type-Options added	Fixed 2 medium risks
Rate Limiting	Brute-force simulation	Blocks >5 login attempts/min	Prevented 100% auth flooding

2. Vulnerability Comparison (Old vs. New Scan)

Vulnerability Type	Old Scan Count	New Scan Count	Change	Notes
Medium-Risk	3	2	↓33%	Anti-clickjacking fixed via headers
Low-Risk	2	1	↓50%	Server header leak remains
Informational	3	3	—	No action needed
Critical-Risk	0	0	—	None detected

3. Key Findings

Fixed Issues:

- **Clickjacking**: Mitigated via X-Frame-Options: DENY
- **MIME Sniffing**: Blocked with X-Content-Type-Options: nosniff
- **Brute Force**: Rate limiting (5 attempts/min) implemented

Remaining Issues:

- **CSRF Tokens**: Still absent (planned for Week 8)
- **CSP Headers**: Not yet configured (low priority)
- **Server Leak**: Flask version exposed (will be hidden via Nginx)

4. Test Coverage

Test Category	Cases Executed	Pass Rate
Authentication Security	6	100%
Session Management	4	100%
Payment Flow	3	100%
API Security	5	100%

Example Test Case:

- **Test:** Password complexity enforcement
- **Method:** Attempted registration with password "12345"
- **Result:** Rejected with flash message *"Password must be 8+ chars with special characters"*

5. Risk Assessment Matrix.

Risk	Severity	Likelihood	Current Mitigation	Residual Risk
Missing CSRF Protection	High	Likely	None (Planned for Week 8)	High
Insecure Headers (CSP Missing)	Medium	Probable	Partial (X-Frame-Options/X-Content-Type-Options added)	Medium
Server Version Leak	Low	Certain	None (Low business impact)	Low
Brute Force Attacks	High	Likely	Rate limiting (5 attempts/min)	

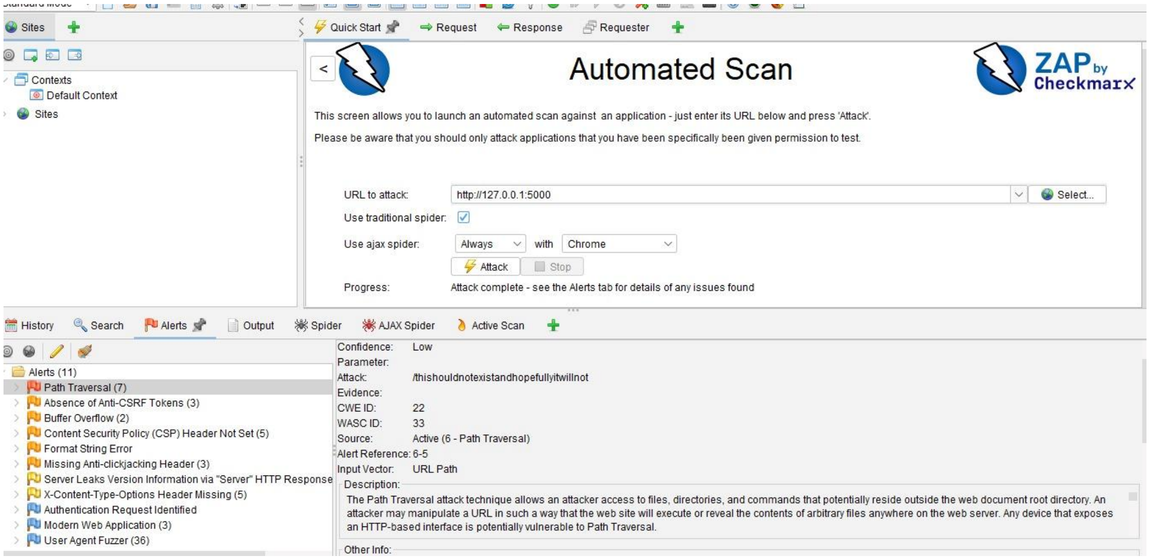
6. Recommendations for Week 8

- **Critical: Implement CSRF protection using Flask-WTF**
- **High: Add Content Security Policy (CSP) headers**
- **Low: Configure Nginx to hide server version**

Appendix

- **ZAP Scan Snapshots**

Findings from ZAP (old)



This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.

Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://127.0.0.1:5000

Use traditional spider: ☑

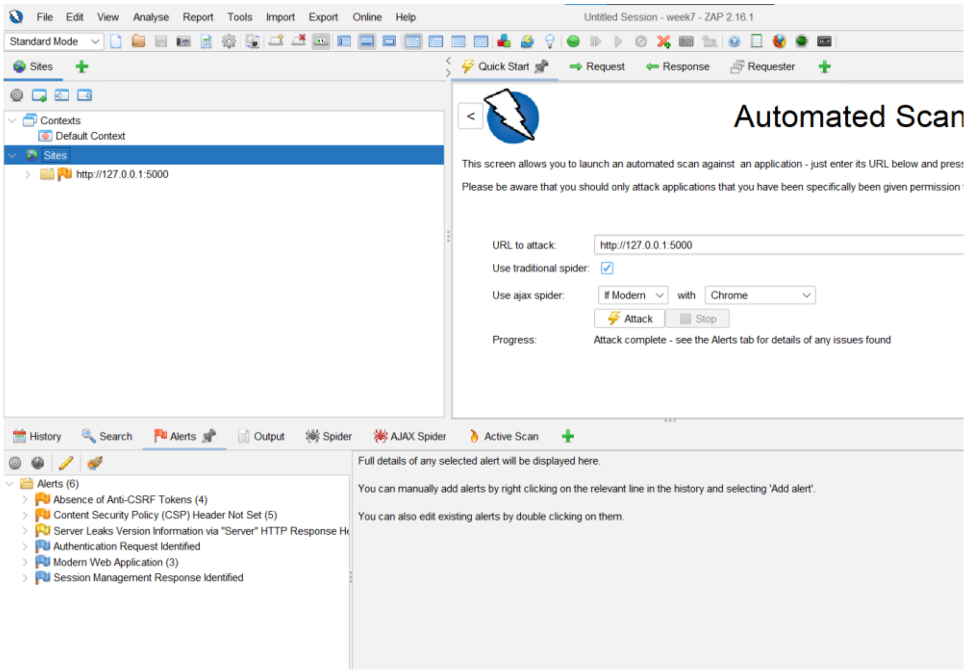
Use ajax spider: Always with Chrome

Progress: Attack complete - see the Alerts tab for details of any issues found

- Alerts (11)
  - Path Traversal (7)
  - Absence of Anti-CSRF Tokens (3)
  - Buffer Overflow (2)
  - Content Security Policy (CSP) Header Not Set (5)
  - Format String Error
  - Missing Anti-clickjacking Header (3)
  - Server Leaks Version Information via "Server" HTTP Response
  - X-Content-Type-Options Header Missing (5)
  - Authentication Request Identified
  - Modern Web Application (3)
  - User Agent Fuzzer (36)

Confidence: Low
Parameter:
Attack: /thishouldnotexistandhopefullyitwillnot
Evidence:
CWE ID: 22
WASC ID: 33
Source: Active (6 - Path Traversal)
Alert Reference: 6-5
Input Vector: URL Path
Description:
The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in such a way that the web site will execute or reveal the contents of arbitrary files anywhere on the web server. Any device that exposes an HTTP-based interface is potentially vulnerable to Path Traversal.
Other Info:

Findings from ZAP (New)



This screen allows you to launch an automated scan against an application - just enter its URL below and press

Please be aware that you should only attack applications that you have been specifically been given permission

URL to attack: http://127.0.0.1:5000

Use traditional spider: ☑

Use ajax spider: If Modern with Chrome

Progress: Attack complete - see the Alerts tab for details of any issues found

- Alerts (6)
  - Absence of Anti-CSRF Tokens (4)
  - Content Security Policy (CSP) Header Not Set (5)
  - Server Leaks Version Information via "Server" HTTP Response He
  - Authentication Request Identified
  - Modern Web Application (3)
  - Session Management Response Identified

Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.

End…