Web Application Security Testing – Project Report

Presented by: Raksha S
Date: 25 July 2025
Tools Used: OWASP ZAP

Project Objective

Conduct security testing on a sample web application.
Generate a detailed report and recommend mitigation strategies.

Tools & Technologies

OWASP ZAP: For automated vulnerability scanning and active attacks.

Target Application:

http://zero.webappsecurity.com

Testing Approach:

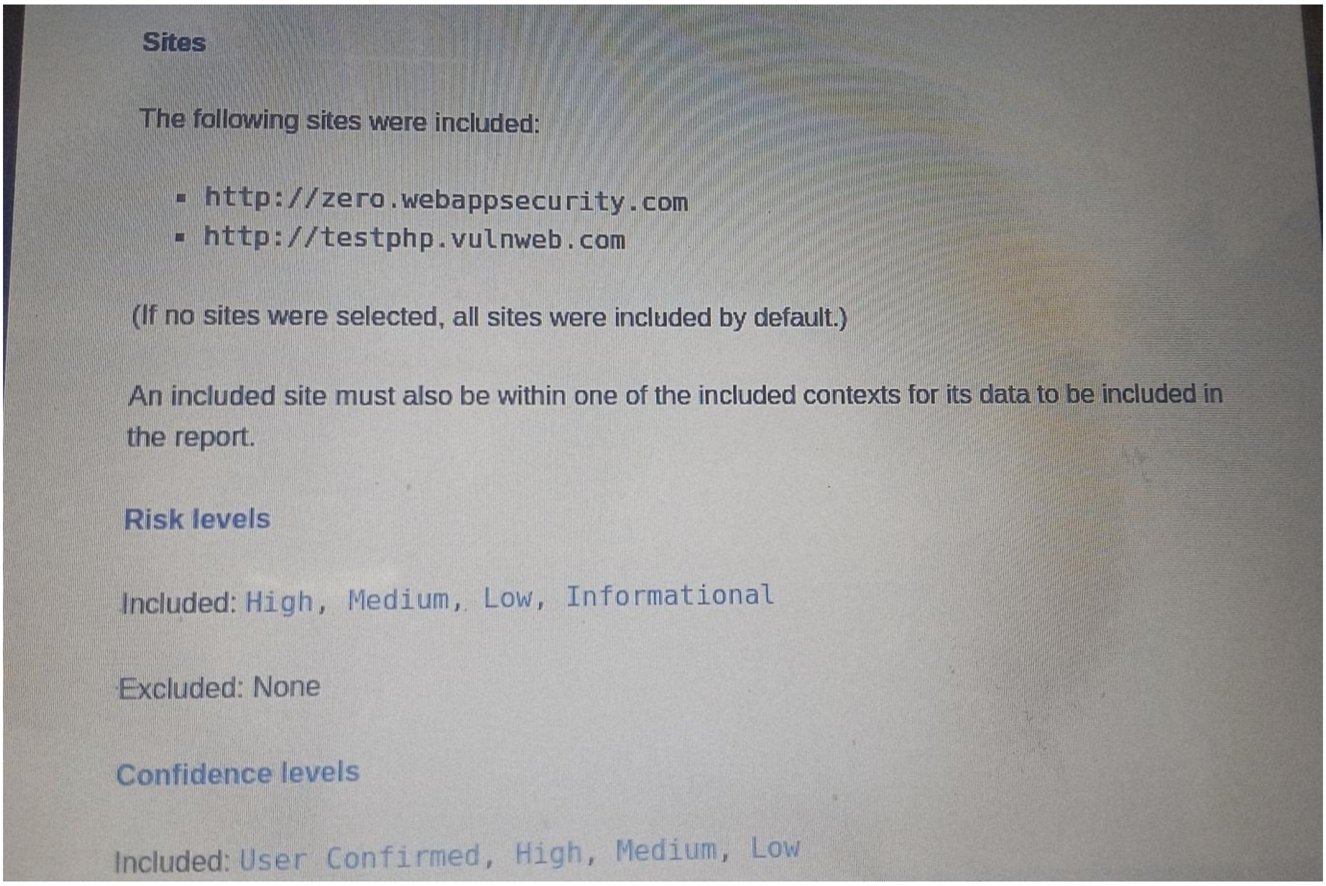
1. Selected the target URL.

2. Crawled the application using Spider and AJAX Spider.

3. Performed Active Scan to identify vulnerabilities.

4. Collected and reviewed the ZAP security report.

5. Analyzed risks and suggested mitigations.



Sites

The following sites were included:

- `http://zero.webappsecurity.com`
- `http://testphp.vulnweb.com`

(If no sites were selected, all sites were included by default.)

An included site must also be within one of the included contexts for its data to be included in the report.

Risk levels

Included: `High`, `Medium`, `Low`, `Informational`

Excluded: None

Confidence levels

Included: `User Confirmed`, `High`, `Medium`, `Low`

ZAP Report Summary

Results Overview:

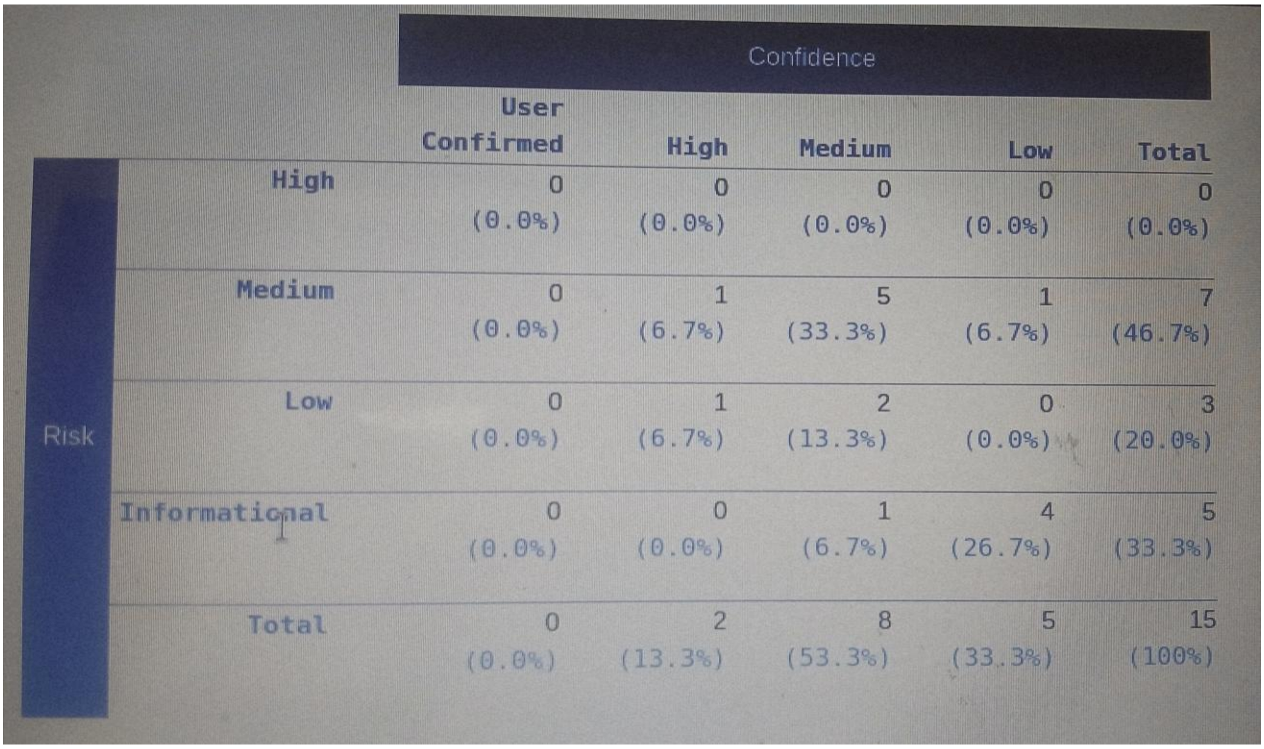
Total Alerts: 15

Medium Risk: 7 alerts

Low Risk: 3 alerts

Informational: 5 alerts

High Risk: 0 alerts



		Confidence				
		User Confirmed	High	Medium	Low	Total
Risk	High	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
	Medium	0 (0.0%)	1 (6.7%)	5 (33.3%)	1 (6.7%)	7 (46.7%)
	Low	0 (0.0%)	1 (6.7%)	2 (13.3%)	0 (0.0%)	3 (20.0%)
	Informational	0 (0.0%)	0 (0.0%)	1 (6.7%)	4 (26.7%)	5 (33.3%)
	Total	0 (0.0%)	2 (13.3%)	8 (53.3%)	5 (33.3%)	15 (100%)

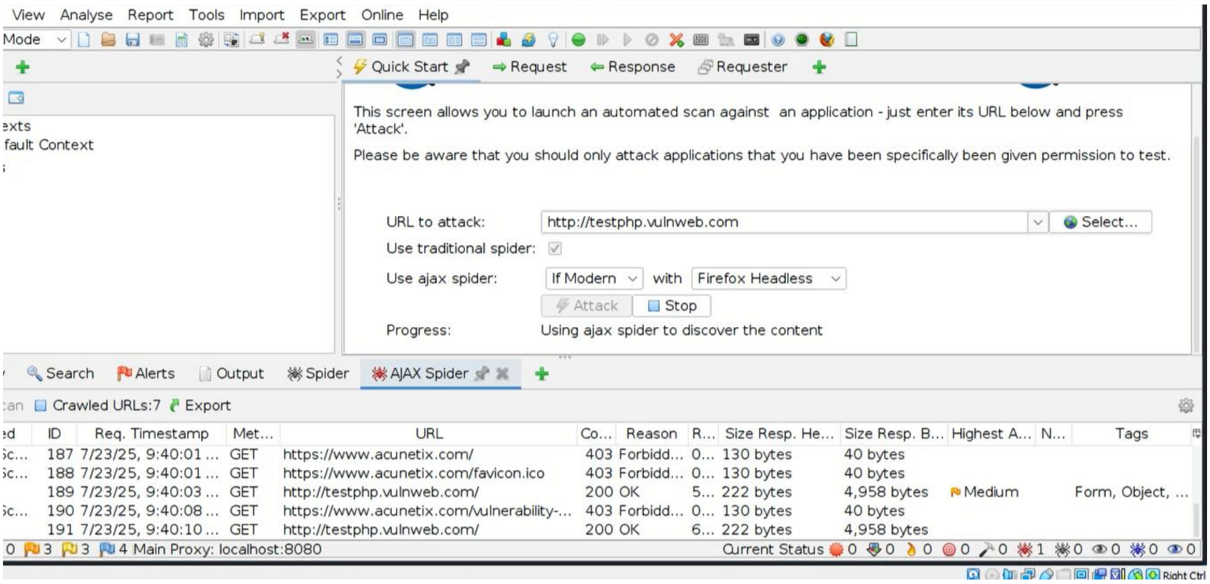
ZAP risk-confidence

Vulnerabilities Identified

Common Issues Found:

1. Missing X-Frame-Options Header
2. Autocomplete Enabled on Password Fields
3. Cookie without Secure and Http Only flags
4. Content-Type sniffing not prevented
5. Information disclosure via headers

I used OWASP ZAP’s AJAX Spider to scan the test website http://testphp.vulnweb.com



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://testphp.vulnweb.com Select...

Use traditional spider: ☑

Use ajax spider: If Modern with Firefox Headless

Attack Stop

Progress: Using ajax spider to discover the content

Crawled URLs:7 Export

ed	ID	Req. Timestamp	Met...	URL	Co...	Reason	R...	Size Resp. He...	Size Resp. B...	Highest A...	N...	Tags
Sc...	187	7/23/25, 9:40:01 ...	GET	https://www.acunetix.com/	403	Forbidd...	0...	130 bytes	40 bytes			
Sc...	188	7/23/25, 9:40:01 ...	GET	https://www.acunetix.com/favicon.ico	403	Forbidd...	0...	130 bytes	40 bytes			
	189	7/23/25, 9:40:03 ...	GET	http://testphp.vulnweb.com/	200	OK	5...	222 bytes	4,958 bytes	Medium		Form, Object, ...
Sc...	190	7/23/25, 9:40:08 ...	GET	https://www.acunetix.com/vulnerability-...	403	Forbidd...	0...	130 bytes	40 bytes			
	191	7/23/25, 9:40:10 ...	GET	http://testphp.vulnweb.com/	200	OK	6...	222 bytes	4,958 bytes			

0 3 3 4 Main Proxy: localhost:8080 Current Status 0 0 0 0 0 1 0 0 0 0

The tool automatically crawls the site to discover all reachable URLs, including dynamic content loaded via JavaScript.

This helps identify hidden endpoints and prepares the application for further vulnerability testing.

The scan shows both successful and restricted responses, indicating access controls and available resources.

Summary:

Successfully performed web application security testing using OWASP ZAP.

Discovered multiple vulnerabilities in the sample app.

Gained hands-on experience in vulnerability scanning and report generation.

Learned the importance of secure coding and mitigation planning.