Experiment No. 6

Aim: To conduct Dynamic Application Security Testing (DAST) using OWASP ZAP for vulnerability scanning.

TOOLS:

- OWASP ZAP (Zed Attack Proxy): An open-source web application security scanner.
- A Target Web Application: A test application running locally (e.g., DVWA - Damn Vulnerable Web Application, or a simple custom-built app). Note: Do not scan live websites you do not own or have explicit permission to test.

Learning Objective: The student should have the ability to understand and perform dynamic security testing.

Theory:

Dynamic Application Security Testing (DAST) is a "black-box" security testing methodology used to find vulnerabilities in a running web application. Unlike static analysis (SAST) which examines source code, DAST interacts with the application from the outside, just as an attacker would, without any knowledge of the internal code or architecture.

The primary goal of DAST is to simulate real-world attacks to identify security flaws such as:

- SQL Injection (SQLi)
- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery (CSRF)
- Broken Authentication and Session Management
- Security Misconfigurations
- Sensitive Data Exposure

OWASP ZAP (Zed Attack Proxy) is a powerful and widely-used open-source tool for DAST. It functions as a "man-in-the-middle" proxy, intercepting traffic between a web browser and the target application. This allows ZAP to inspect requests and responses, manipulate them, and launch automated attacks to discover security weaknesses.

Procedure / Implementation:

This procedure outlines how to perform a basic vulnerability scan using OWASP ZAP.

1. Installation and Setup

- Download and install OWASP ZAP from the official website: https://www.zaproxy.org/download/.
- Ensure your target web application is running and accessible from your browser (e.g., http://localhost:8080).

2. Automated Scan (Quick Start) This is the simplest way to scan a web application.

- Launch OWASP ZAP.
- In the **Quick Start** tab, locate the **Automated Scan** section.
- Enter the full URL of your target application in the "URL to attack" field (e.g., http://localhost:8080).
- Click the **Attack** button.
- ZAP will first perform "spidering" to discover all pages and links, and then it will launch an "active scan" to attack the discovered pages and parameters.

3. Analyzing the Results

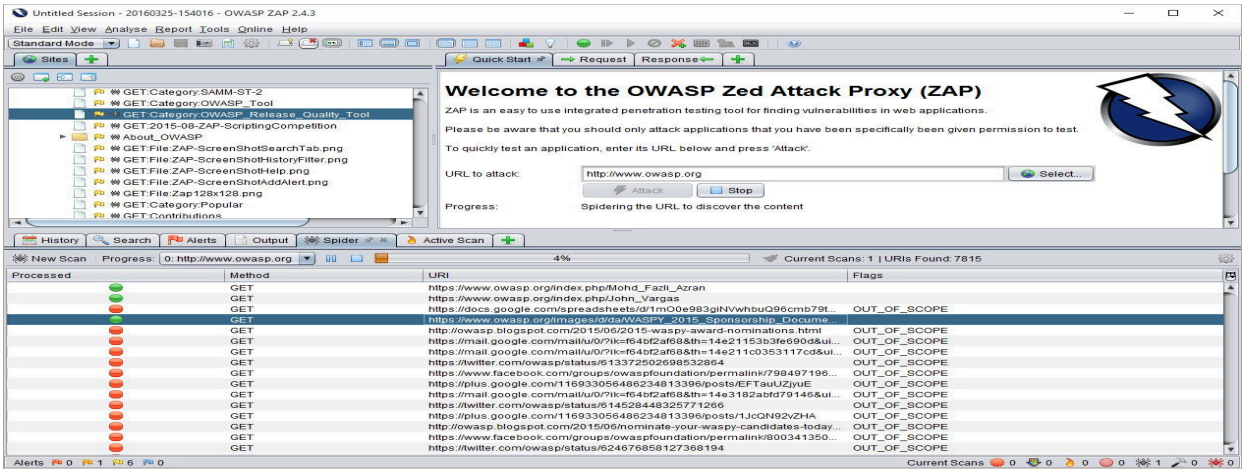
- Once the scan is complete, click on the **Alerts** tab at the bottom of the ZAP window.
- Here you will find a list of all potential vulnerabilities discovered, categorized by risk level (High, Medium, Low, Informational).
- Clicking on any alert will display detailed information in the right-hand panel, including the affected URL, the parameter, a description of the vulnerability, and a suggested solution.

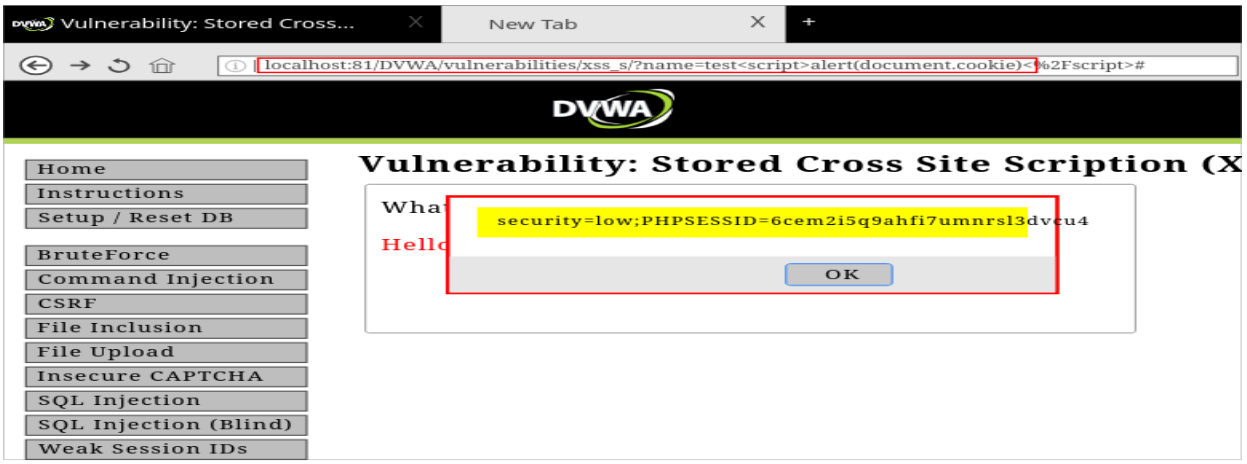
4. Generating a Report A formal report is essential for documenting and sharing findings.

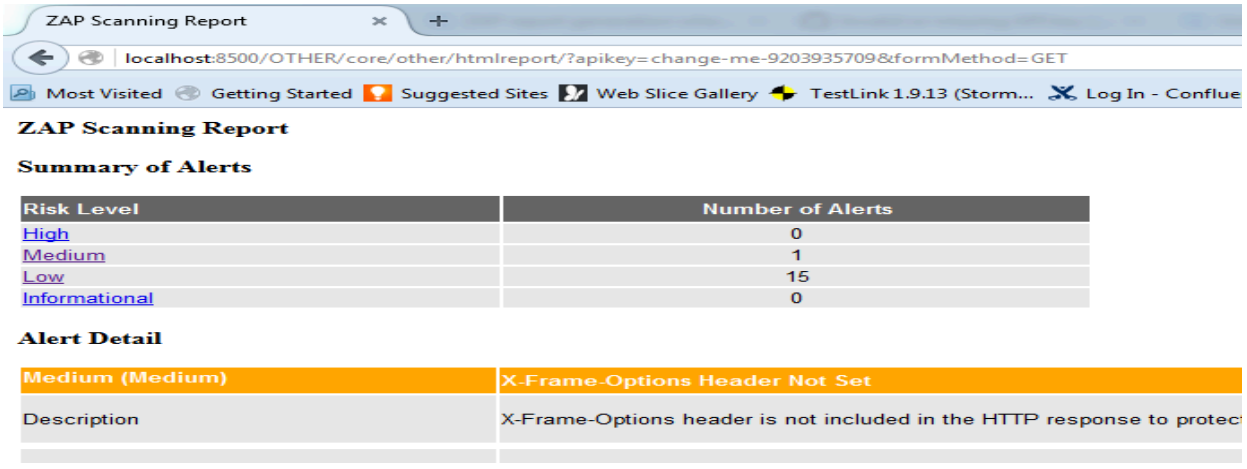
- From the top menu bar, navigate to **Report -> Generate Report...**.
- Choose a title, select a template (e.g., **HTML Report**), and choose a location to save the file.
- Click **Generate**.

- Open the generated HTML file in a browser to view the comprehensive security report.

Output:







Learning Outcome: The student should have the ability to understand and perform dynamic security testing.

- LO1: To understand the fundamental principles of Dynamic Application Security Testing (DAST) as a black-box testing methodology.
- LO2: To use OWASP ZAP to configure and execute an automated vulnerability scan against a running web application.

Course Outcome: Upon completion of the course, students will be able to integrate DAST into the software development lifecycle to proactively identify and remediate security vulnerabilities in running web applications, thereby reducing the risk of cyber attacks.

Conclusion: This experiment provided practical, hands-on experience with Dynamic Application Security Testing using the industry-standard tool, OWASP ZAP. By performing an automated scan, students successfully identified potential security vulnerabilities in a running web application, analyzed the detailed alerts, and generated a professional security report. The experiment highlights the critical role of DAST in a comprehensive security strategy, as it effectively simulates real-world attacks to uncover flaws that might not be apparent from source code analysis alone.

Name:
Class: BE-CSE
Roll No.:

For Faculty Use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				