**LAB 2 : VULNERABILITY SCANNING & RISK ASSESSMENT**

**OWASP ZAP (Zed Attack Proxy)** to scan a web app and detect common **web vulnerabilities** such as:

1. **Cross-Site Scripting (XSS)**
2. **SQL Injection**
3. **Security Misconfiguration or Information Disclosure**

Step 1: Setup OWASP ZAP

Open **terminal** and type as below:

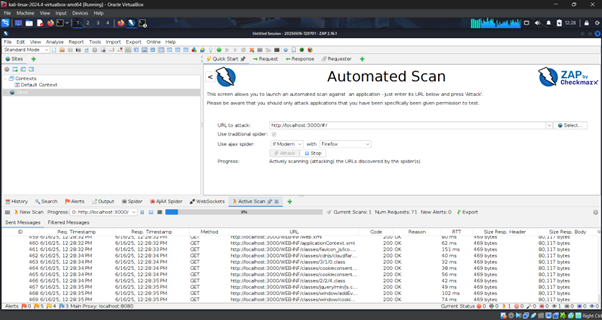
sudo apt update

sudo apt install zaproxy

Step 2: Choose a Target Web Application : open the website at firefox

Step 3: Run an Active Scan

1. In ZAP, go to **Quick Start > Automated Scan**.
2. Enter the **target URL** (e.g., http://localhost/dvwa/).
3. Click **Attack** — ZAP will spider (crawl) the site and perform an **Active Scan**.

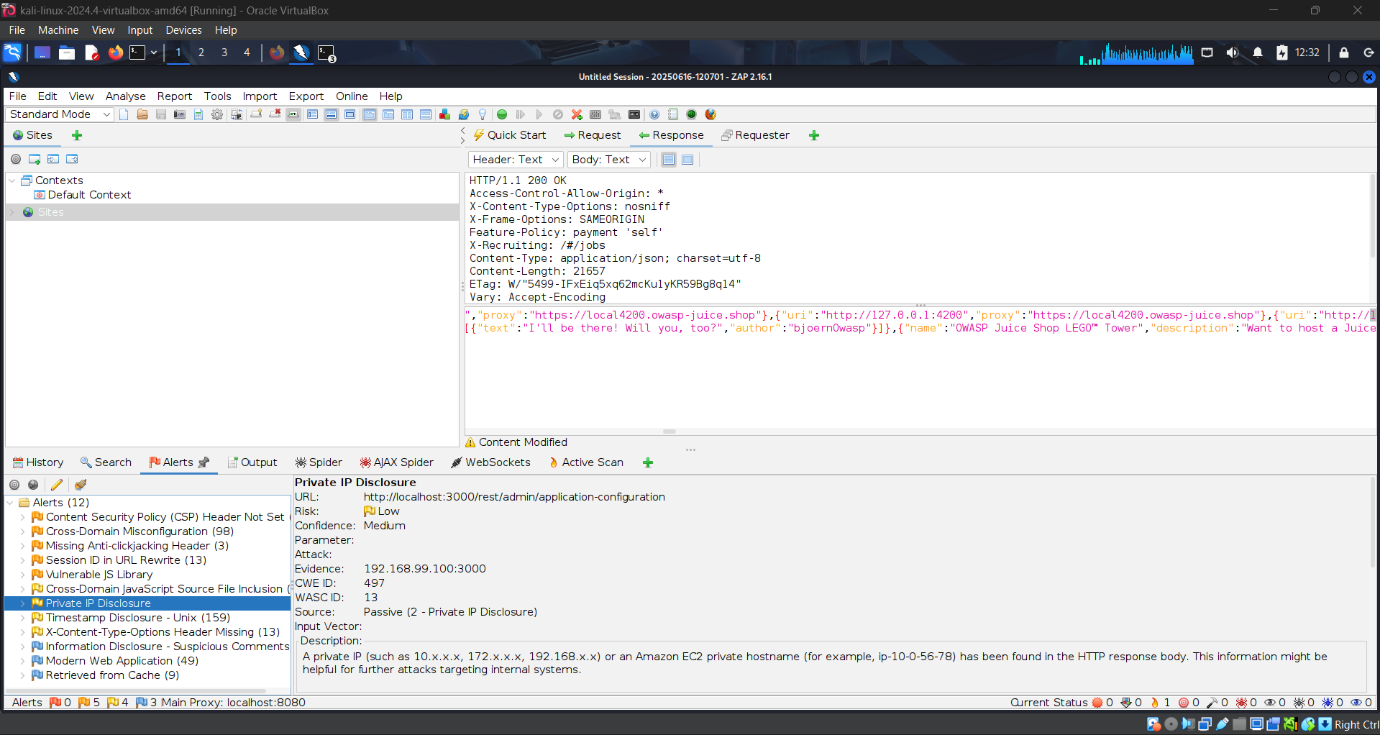


If DVWA requires you to **login first in browser**, then **start the scan with an active session**. Use the **browser with ZAP's proxy**.

Step 4: Review Vulnerabilities Found

Once the scan is complete, look at the **"Alerts" tab** on the left. ZAP will categorize findings like:

**Example Vulnerabilities You Might Find:**

****

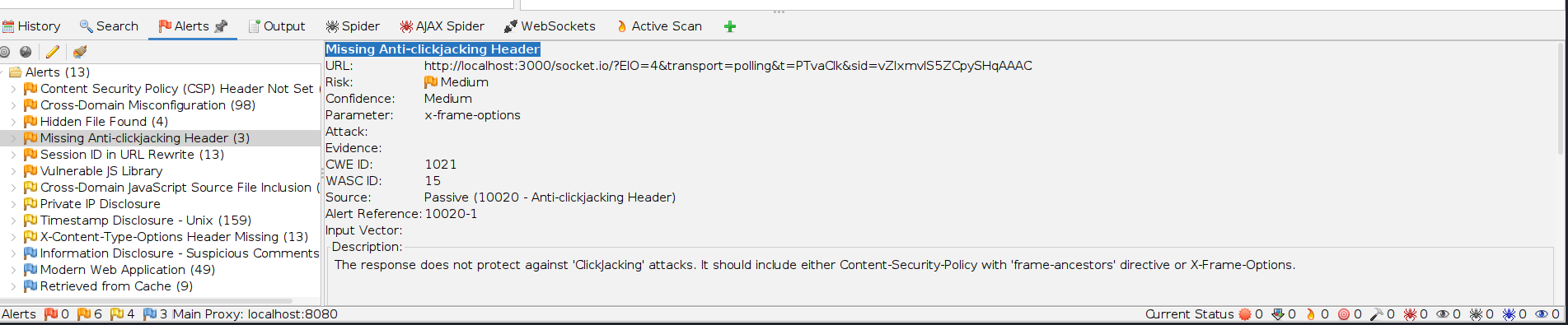
| **Vulnerability** | **Description** |
| --- | --- |
| 🔴 **XSS (Reflected or Stored)** | Unsanitized input echoed in response — attacker can inject scripts. |
| 🔴 **SQL Injection** | Input fields not sanitized — attacker can manipulate SQL queries. |
| 🟠 **Information Disclosure** | Server leaks headers, comments, or error messages. |
| 🟡 **Security Headers Missing** | Common HTTP headers for security (e.g., CSP, X-Frame-Options) are not present. |

**📝 Step 5: Report Your Findings**

**HOW TO REPORT:**

|  |  |  |  |
| --- | --- | --- | --- |
| Tool Used: | : |  | |
| Target url: | : |  |
| Date: | : |  |
| Vulnerability name:  [Once the scan is complete, look at the **"Alerts" tab** on the left at ZAP] | : |  |
| Brief explanation about vulnerability: | : |  |
| CVE ID: | : |  |
| OWASP Top 10 topping: | : |  |
| Risk level classification: | : |  |

Understanding **CVE IDs** and **OWASP Top 10 mapping** is key when analysing vulnerabilities.



# LOOK AT THE ALERTS

**EXAMPLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| Tool Used: | : | OWASP ZAP (Zed Attack Proxy) | |
| Target url: | : | http://localhost:3000 |
| Date: | : | 16 June 2025 |
| Vulnerability name:  [Once the scan is complete, look at the **"Alerts" tab** on the left at ZAP] | : | Missing Anti-clickjacking Header |
| Brief explanation about vulnerability: | : | This vulnerability occurs when a web application does not include an X-Frame-Options or Content-Security-Policy: frame-ancestors header. Without this header, the site can be embedded in an <iframe> by a malicious site, making it vulnerable to Clickjacking attacks, where users can be tricked into clicking elements they don't see or intend to click. |
| CVE ID: | : | There is no specific CVE for this generic issue, but it's a known misconfiguration. |
| OWASP Top 10 topping: | : | A05:2021 – Security Misconfiguration |
| Risk level classification: | : | Low (but can be elevated based on business impact or combined with other vulnerabilities) |

**1. What is a CVE ID?**

**CVE** stands for **Common Vulnerabilities and Exposures**. A **CVE ID** is a unique identifier for a publicly known cybersecurity vulnerability.

**Example:**

* **CVE-2021-44228** — This refers to the famous **Log4Shell** vulnerability in Apache Log4j.

**2. What is OWASP Top 10 Mapping?**

OWASP Top 10 is a list of the 10 most critical web application security risks, updated every few years by the OWASP Foundation.

Latest OWASP Top 10 (2021 version):

Code Category

A01 Broken Access Control

A02 Cryptographic Failures

A03 Injection

A04 Insecure Design

A05 Security Misconfiguration

A06 Vulnerable & Outdated Components

A07 Identification & Authentication Failures

A08 Software & Data Integrity Failures

A09 Security Logging & Monitoring Failures

A10 Server-Side Request Forgery (SSRF)

**Mapping Example:**

Let’s say you find SQL Injection using OWASP ZAP.

✅ CVE Mapping: You find this vulnerability exists in a known CMS and has a CVE like CVE-2018-12345.

✅ OWASP Mapping: SQL Injection maps to OWASP A03 – Injection.

Once done Zap, refer at Alerts.

Copy any alert name at ZAP, and at chat ask [what is the CVE ID and OWASP Top 10 for Cross site scripting (DOM based)]

ZAP shows as below:

**1. CWE ID – Common Weakness Enumeration**

* Created by **MITRE**
* Describes software **weakness types** (e.g., buffer overflows, improper input validation)

For **DOM-Based XSS**, ZAP usually shows:

**✅ CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')**

* This is the general CWE for all types of XSS, including reflected, stored, and DOM-based.

**🔹 2. WASC ID – Web Application Security Consortium Threat Classification**

* An older classification system still used by ZAP and some tools.
* Each ID maps to a specific web vulnerability class.

For **DOM-Based XSS**, ZAP often shows:

**✅ WASC-8: Cross-Site Scripting**

|  |
| --- |
| Vulnerability: DOM-Based Cross-Site Scripting (XSS)  Description: Unsanitized user input is read and processed by browser-side JavaScript, allowing attacker-controlled script execution.  CWE ID: CWE-79 – Improper Neutralization of Input During Web Page Generation  WASC ID: WASC-8 – Cross-Site Scripting  OWASP Top 10 Mapping: A03 – Injection (OWASP 2021)  CVE ID: Not applicable (vulnerability found in custom test app)  Risk: Allows attacker to execute malicious JavaScript in victim's browser. |