OWASP ZAP

Introduction to OWASP

(Open Web Application Security Project)

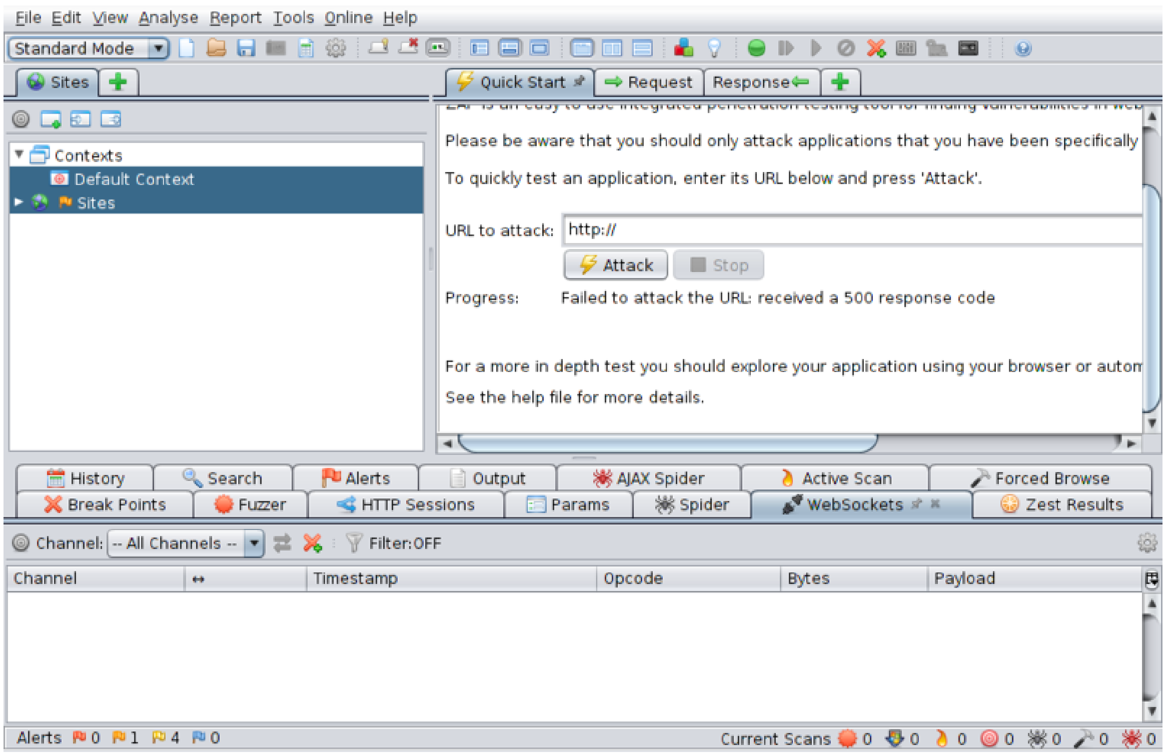
- OWASP is a non-profit organisation that focuses on improving the security of software.
- They provide open-source tools, frameworks, and educational resources to help developers create secure applications.
- One of their most well-known projects is the *OWASP Top 10*, which lists the most critical web application vulnerabilities. This list acts as a guide for developers, security professionals, and organisations to understand and prioritise the risks associated with insecure software.

OWASP Top Ten | OWASP Foundation
The OWASP Top 10 is the reference standard for the most critical web application security risks. Adopting the OWASP Top 10 is perhaps the most effective first step
https://owasp.org/www-project-top-ten/

▼ What is OWASP ZAP?

- OWASP ZAP (Zed Attack Proxy) is an open-source security testing tool for web applications.
- It helps developers and security professionals find vulnerabilities in web applications through both passive and active scanning.
- It is one of the most widely used tools for dynamic application security testing (DAST).
- **Features**:
    - **Active and Passive Scanning**: Can passively observe traffic and actively test for vulnerabilities by simulating malicious behaviour.
    - **Automated & Manual Testing**: Works in both fully automated modes and allows for manual exploration of web applications.
    - **Authentication Handling**: Supports scans behind login screens.
    - **Proxy Functionality**: Can intercept and modify traffic to simulate attack scenarios.

▼ OWASP ZAP Architecture & Key Components



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

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://

Progress: Failed to attack the URL: received a 500 response code

For a more in depth test you should explore your application using your browser or autom

See the help file for more details.

Components:

- **Menu Bar**: For managing sessions, creating reports, and accessing tools.
- **Toolbar**: Provides shortcuts to ZAP's most common features.
- **Tree Window**: Displays the hierarchical view of the site being tested.
- **Workspace Window**: Displays requests, responses, and site information.
- **Alerts Tab**: Shows security alerts and their severity.
- **History Tab**: Logs all HTTP requests and responses.

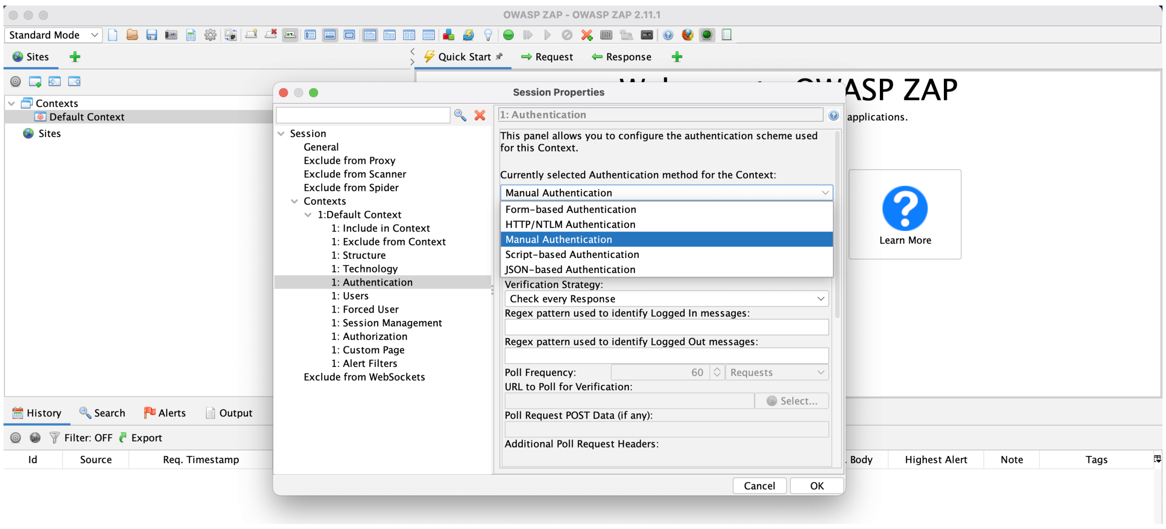
▼ Types of Scanning in ZAP

Passive Scanning:

- Passively scans HTTP traffic for vulnerabilities without interacting with the application.
- Ideal for continuous monitoring and less disruptive, but limited in depth.

Active Scanning:

- Actively interacts with the web application by sending modified requests to identify vulnerabilities like SQL injection and XSS.
- Can be disruptive to live systems and should be used in a test environment.



This panel allows you to configure the authentication scheme used for this Context.

Currently selected Authentication method for the Context: Manual Authentication

Verification Strategy: Check every Response

Regex pattern used to identify Logged In messages:

Regex pattern used to identify Logged Out messages:

Poll Frequency: 60 Requests

URL to Poll for Verification:

Poll Request POST Data (if any):

Additional Poll Request Headers:

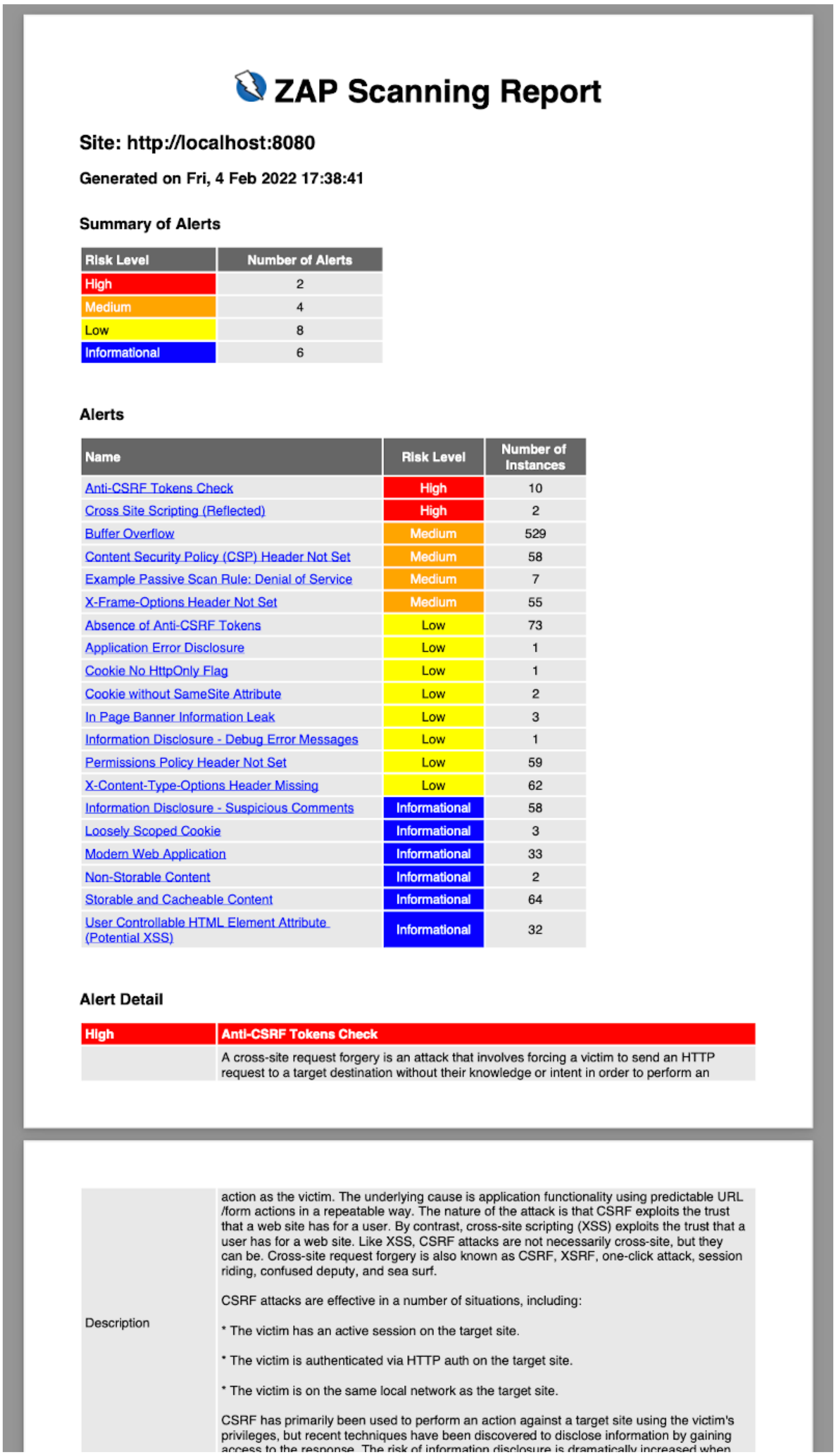
▼ Demonstrating OWASP ZAP with Juice Shop:

Juice Shop is a vulnerable web application designed for security training and testing, making it an ideal target for ZAP's features.

Here’s how you can run the demo:

1. **Access the Juice Shop demo**:
    - Visit the publicly hosted instance of Juice Shop to access the application directly in your browser.
2. **Launch OWASP ZAP**:
    - Start OWASP ZAP locally and set it up as a proxy.
    - Configure your browser to pass all traffic through ZAP (set the proxy to `localhost:8080`).
3. **Perform a passive scan**:
    - Open the Juice Shop application in your browser. ZAP will automatically capture the traffic and list it in the interface.
    - This passive scan will give you insights into potential security issues without actively attacking the site.
4. **Active scan**:
    - To perform a more thorough analysis, right-click on the Juice Shop entry in ZAP’s “Sites” tree, and select "Attack" > "Active Scan".
    - This will allow ZAP to probe the application for vulnerabilities like SQL injection, cross-site scripting (XSS), and broken authentication.
5. **Explore vulnerabilities**:

    ZAP will generate a report highlighting security issues detected, which can include a variety of web vulnerabilities like XSS, injection flaws, sensitive data exposure, and more.



ZAP Scanning Report

Site: http://localhost:8080

Generated on Fri, 4 Feb 2022 17:38:41

Summary of Alerts

Risk Level	Number of Alerts
High	2
Medium	4
Low	8
Informational	6

Alerts

Name	Risk Level	Number of Instances
Anti-CSRF Tokens Check	High	10
Cross Site Scripting (Reflected)	High	2
Buffer Overflow	Medium	529
Content Security Policy (CSP) Header Not Set	Medium	58
Example Passive Scan Rule: Denial of Service	Medium	7
X-Frame-Options Header Not Set	Medium	55
Absence of Anti-CSRF Tokens	Low	73
Application Error Disclosure	Low	1
Cookie No HttpOnly Flag	Low	1
Cookie without SameSite Attribute	Low	2
In Page Banner Information Leak	Low	3
Information Disclosure - Debug Error Messages	Low	1
Permissions Policy Header Not Set	Low	59
X-Content-Type-Options Header Missing	Low	62
Information Disclosure - Suspicious Comments	Informational	58
Loosely Scoped Cookie	Informational	3
Modern Web Application	Informational	33
Non-Storable Content	Informational	2
Storable and Cacheable Content	Informational	64
User Controllable HTML Element Attribute (Potential XSS)	Informational	32

Alert Detail

High	Anti-CSRF Tokens Check
Description	A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL /form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when

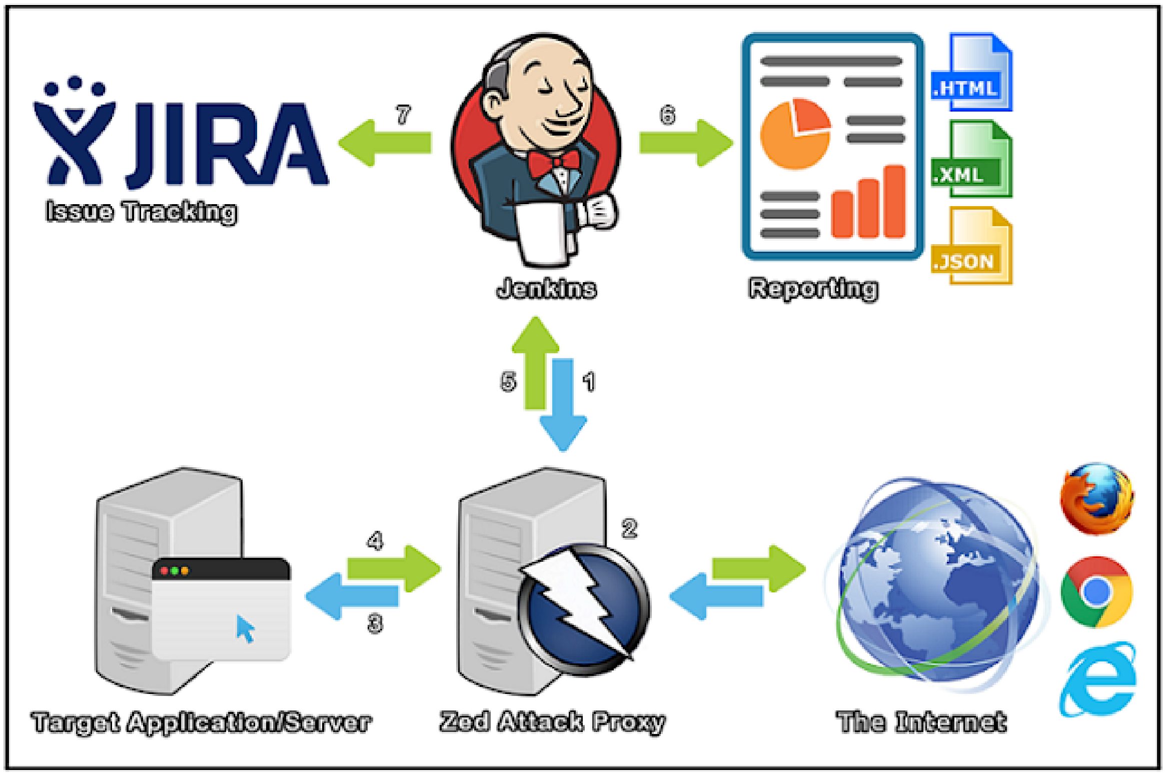
▼ Integration with CI/CD Pipelines

Why Integrate ZAP with CI/CD?:

- Ensures security tests are run continuously, reducing the risk of introducing vulnerabilities into production environments.
- ZAP can be triggered as part of the build process, automating vulnerability detection.

Tools & Integration:

- ZAP works with Jenkins, GitLab CI, and other CI tools.
- Generate reports automatically and track vulnerabilities over time.



▼ Authentication Mechanisms in ZAP

- **Form-based Authentication:** ZAP can handle simple login forms and simulate the login process, maintaining the session throughout the scan. This is useful for web applications with basic login setups.
- **Basic & Digest Authentication:** Credentials (username and password) are automatically handled in the request headers for these simpler

HTTP-based authentication methods, allowing ZAP to seamlessly test protected resources.

- **OAuth & SAML Authentication:** ZAP can work with modern authentication mechanisms like OAuth2 and SAML, often used in APIs and Single Sign-On (SSO) systems. While these require more complex handling, custom scripting enables token-based workflows.
- **Authentication Scripting:** ZAP offers custom scripts for multi-step or complex authentication flows, like 2FA or dynamic token exchange, ensuring the session remains active during scans.

▼ ZAP Spidering in Detail

- **Crawling Depth & Limitations:** ZAP's spidering can be configured to set the depth of crawling, controlling how many levels deep the spider explores the application's structure. This is useful for larger applications where deep crawling might result in overload or performance issues.
- **Handling JavaScript-Rich Applications:** Traditional spiders may miss pages rendered dynamically via JavaScript. ZAP's **AJAX Spider** addresses this issue by executing JavaScript during the crawl to capture dynamic content.
- **Exclusions & Constraints:** ZAP allows for the exclusion of certain URLs (like logout links or admin panels) to avoid unnecessary or harmful actions during spidering, such as logging out users or accessing sensitive sections.

▼ Advanced API Security Testing with ZAP

- **REST and SOAP API Testing:** ZAP supports testing for both RESTful APIs and SOAP services. It automatically identifies endpoints and can send requests to test for vulnerabilities like improper input validation or exposed sensitive data.
- **Automatic and Manual Testing:** While ZAP can automate scans on API endpoints, more complex testing—like multi-step API calls or chained responses—might require manual testing or the use of custom scripts.
- **Custom Headers & Tokens:** Many APIs rely on custom headers for authentication (e.g., **JWT**, **Bearer tokens**). ZAP allows testers to configure these headers for accurate scans of API endpoints.

▼ HTTP Interception and Modification

- **Intercept and Modify Requests:** ZAP's proxy allows testers to intercept HTTP/HTTPS requests, modify them, and replay the altered request. This helps simulate attacks such as **parameter tampering**, **SQL injection**, or altering cookies.
- **Session Management Testing:** ZAP's interception capabilities allow you to test how a web application manages session cookies or tokens, which can expose vulnerabilities like **session fixation** or **token replay** attacks.
- **Replay Modified Requests:** After intercepting and modifying HTTP requests, testers can replay them to see if the application properly handles unexpected inputs, like altered GET/POST parameters.

▼ Extending ZAP with Add-ons

- **Custom Add-ons for Specific Vulnerabilities**: ZAP's add-on marketplace includes plugins for testing specific vulnerabilities like **Cross-Site Scripting (XSS)**, **SQL Injection (SQLi)**, and **Content Security Policies (CSP)**.
- **Framework-Specific Scanning**: ZAP add-ons extend its scanning capabilities to handle modern frameworks like **Angular**, **React**, or **Vue.js**, allowing for more precise vulnerability detection in applications using JavaScript-based front ends.
- **Enhanced Reporting**: Add-ons can enhance ZAP's default reporting tools, enabling more detailed or customised reports that suit different security standards (e.g., OWASP Top 10, PCI-DSS).

▼ Customisation and Scripting in ZAP

- **JavaScript & Groovy Scripting**: ZAP allows testers to write custom scripts in JavaScript, Python, or Groovy for advanced testing scenarios, such as simulating race conditions or testing specific business logic flaws.
- **Advanced Vulnerability Tests**: By writing custom scripts, testers can go beyond default ZAP scans and target niche vulnerabilities that may not be covered by ZAP's built-in rules, such as **business logic flaws** or **custom encryption weaknesses**.