

OWASP ZAP

Web Application Vulnerability Scanner



Kushal Goswami

PTID-CHE-SEP-25-219

Introduction

Why Vulnerability Management Matters

Vulnerability management is the proactive discipline of finding and fixing security flaws before attackers can exploit them. In today's threat landscape, regular scanning is essential to detect emerging vulnerabilities, reduce organizational risk, and maintain a robust security posture.

OWASP ZAP stands out as a powerful, open-source solution offering comprehensive scanning capabilities with detailed, actionable reporting that security teams can trust.



About OWASP ZAP



Open-Source Security

OWASP ZAP (Zed Attack Proxy) is a community-driven, open-source tool designed specifically for testing web application security vulnerabilities.



Comprehensive Detection

Identifies critical vulnerabilities including missing security headers, insecure cookie configurations, SQL injection, XSS, and more.



Cross-Platform

Completely free and supports Windows, Linux, and macOS environments, making it accessible for any security team.



Flexible Scanning

Offers both automated scanning for rapid assessment and manual exploration modes for deep security testing with detailed reports.

ZAP Testing Workflow

Follow this structured approach to conduct comprehensive web application security assessments using OWASP ZAP.

01

Select Scan Mode

Choose between Automated Scan for quick vulnerability detection or Manual Explore for hands-on testing and discovery

02

Password Testing

Utilize the Fuzzer tool to test password fields against wordlists and identify weak authentication mechanisms

03

Authentication Setup

Configure Form-Based Authentication to enable deeper scanning of authenticated areas and session management

04

Report Generation

Generate comprehensive reports detailing discovered vulnerabilities, severity levels, and recommended remediation steps

Automated Scan

Quick Vulnerability Assessment

The automated scan provides rapid security assessment of web applications with minimal configuration required.

→ Launch Quick Start

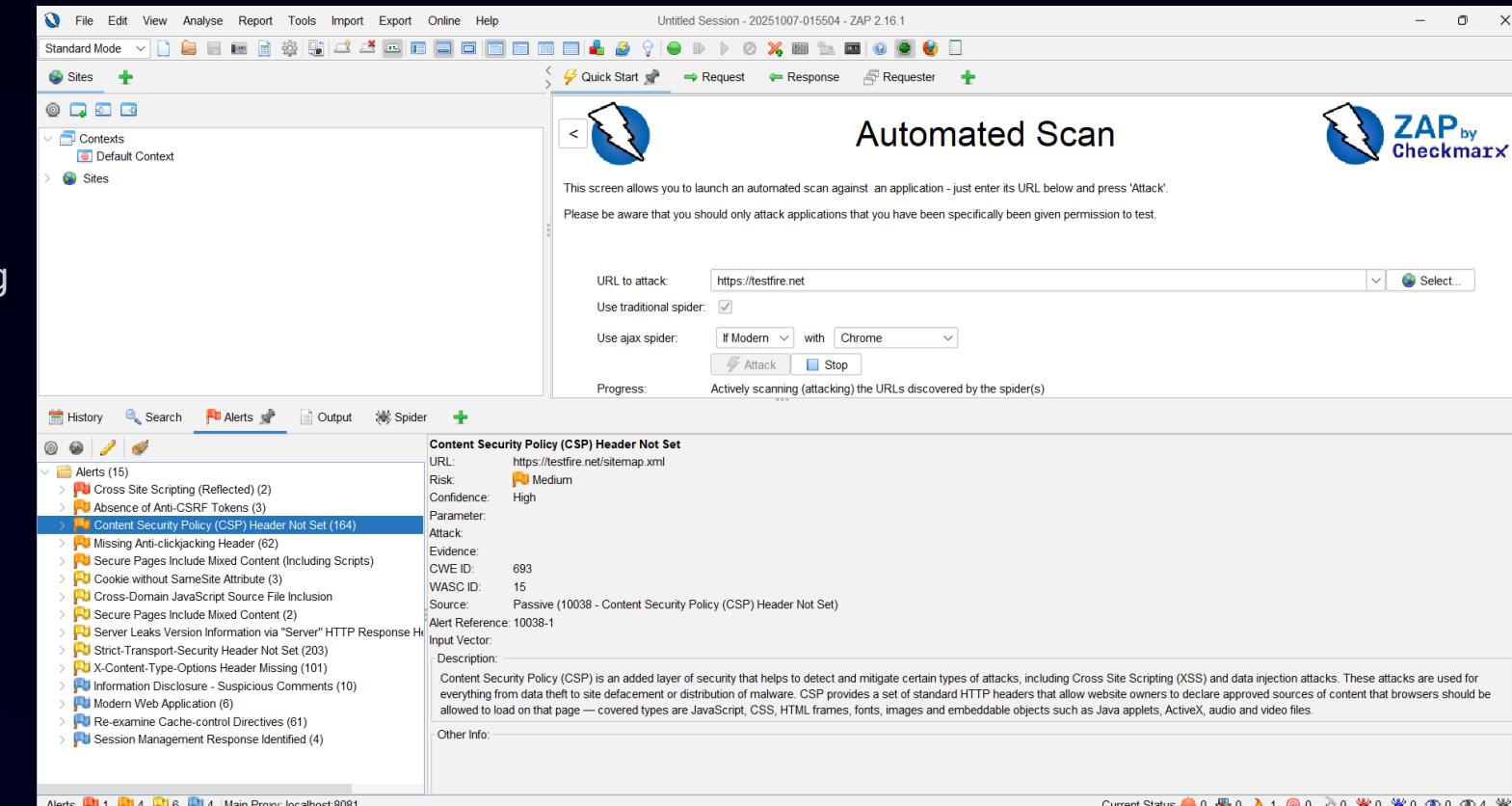
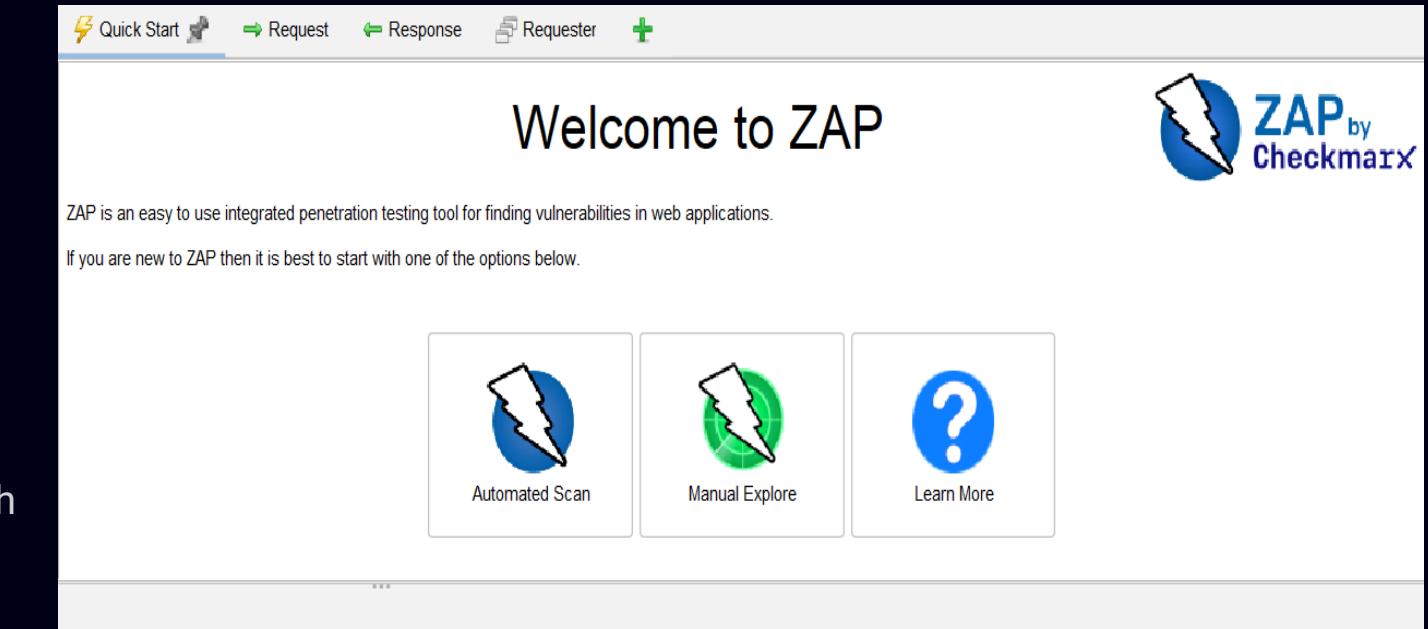
Open Quick Start tab and enter your target URL (e.g., <https://testfire.net>) into the designated field.

→ Initiate Attack

Click the 'Attack' button to begin automatic scanning and crawling of all accessible web pages and endpoints.

→ Review Findings

Navigate to the 'Alerts' tab to examine discovered vulnerabilities, each categorized by severity level (High, Medium, Low).



Manual Explore

Deep Dive into Application Behavior

Manual exploration allows security testers to interact naturally with the application while ZAP captures all HTTP traffic, revealing vulnerabilities that automated scans might miss.

Step 1: Launch Browser

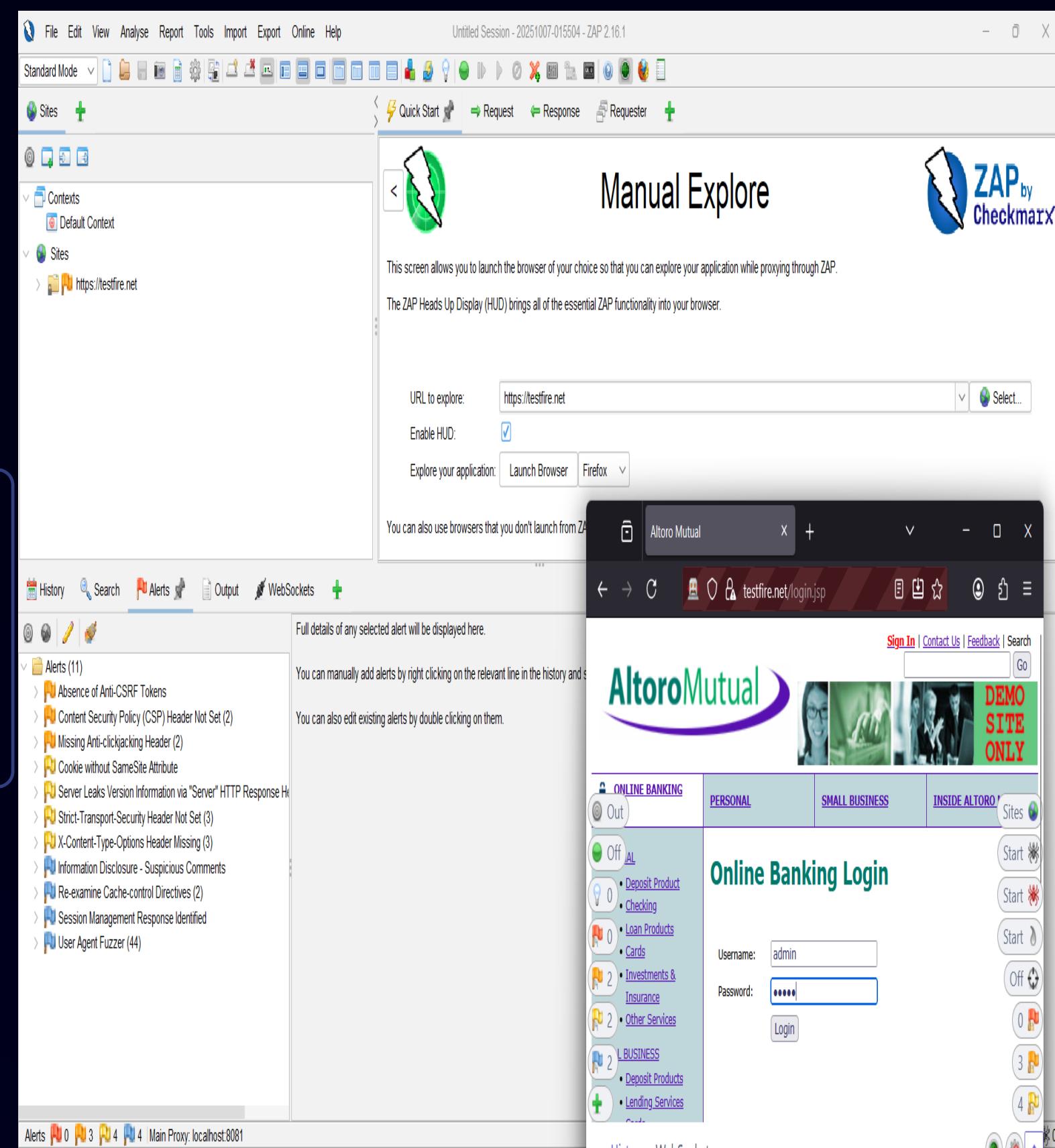
Open Quick Start → Manual Explore → Launch Browser. ZAP will configure a proxied browser instance automatically.

Step 2: Interactive Browsing

Manually navigate through the target site, clicking links, submitting forms, and exploring all functionality while ZAP captures requests and responses in real-time.

Step 3: Test Critical Areas

Focus on testing authentication pages, form submissions, session handling, and hidden sections that require user interaction to access.



Password Cracking (Fuzzer)

The Fuzzer tool in ZAP enables ethical password strength testing by systematically trying different password combinations against login forms to identify weak credentials.

Capture POST Request

Locate and capture the POST request from the login form in the Sites tree panel after attempting a login.

Configure Fuzzer

Right-click the request → Attack → Fuzz → Highlight the password field parameter for targeted fuzzing.

Load Wordlist

Load a password dictionary file (such as `rockyou.txt` or a custom wordlist) containing common passwords to test.

Execute Attack

Start the Fuzzer to systematically test each password. Monitor response codes and sizes to identify successful authentication attempts.

The screenshot captures the ZAP (Zed Attack Proxy) interface during a fuzzing session. The top navigation bar includes File, Edit, View, Analyse, Standard Mode, Sites, and a toolbar with icons for Break, Delete, Exclude from, Run application, New Alert, and Alerts for This Node. The left sidebar displays a tree view of the application's structure, including nodes like GET:PrivacyPolicy, GET:admin, and POST:doLogin(). The main workspace is divided into several tabs: History, Search, Alerts, Output, WebSockets, and Fuzzer. The Fuzzer tab is currently active, showing a request for 'https://testfire.net/doLogin' with various payload fields and a payload editor. A context menu is open over a selected POST request, showing options like Active Scan, AJAX Spider, Client Spider, and Fuzz. The bottom section shows a table of fuzzing results with columns for Task ID, Message Type, Code, Reason, RTT, Header Size, Body Size, Highest Alert, State, and Payloads. The table lists 10 rows of results, mostly 302 Found responses with varying headers and body sizes.

Task ID	Message Type	Code	Reason	RTT	Size Resp. Header	Size Resp. Body	Highest Alert	State	Payloads
1	Fuzzed	302	Found	803 ms	277 bytes	0 bytes			admin, admin
0	Original	302	Found	779 ms	126 bytes	0 bytes	Low		
2	Fuzzed	302	Found	792 ms	126 bytes	0 bytes			admin, user
3	Fuzzed	302	Found	772 ms	126 bytes	0 bytes			admin, password
4	Fuzzed	302	Found	792 ms	126 bytes	0 bytes			password, admin
5	Fuzzed	302	Found	802 ms	126 bytes	0 bytes			password, user
6	Fuzzed	302	Found	752 ms	126 bytes	0 bytes			password, password
7	Fuzzed	302	Found	802 ms	126 bytes	0 bytes			user123, admin
8	Fuzzed	302	Found	792 ms	126 bytes	0 bytes			user123, user
9	Fuzzed	302	Found	790 ms	126 bytes	0 bytes			user123, password

Authenticated Scan

Testing Behind the Login

Authenticated scanning enables ZAP to test vulnerabilities in protected areas of the application that require valid user credentials to access.

- **Create Context**

Define a new context including your target site, then configure Form-Based Authentication with login URL and form parameters.

- **Configure User**

Add a test user account with valid credentials and enable Forced User Mode to maintain automatic re-authentication.

- **Run Authenticated Scan**

Execute both Active Scan and Spider with the authenticated user to achieve comprehensive coverage of protected areas.

- **Compare Results**

Analyze the difference in vulnerability alerts discovered before and after authentication to identify access-control issues.

The screenshot displays the ZAP interface with several windows open:

- Sites Panel:** Shows the target site <https://testfire.net> with various URLs listed under it.
- New Alert... Context Submenu:** A context submenu is open under the "New Alert..." option, showing options like "Include in Context", "Default Context", "New Context", etc.
- Session Properties Panel:** Configures the authentication scheme for the context, selecting "Form-based Authentication" with a target URL of <https://testfire.net/doLogin>.
- Add a New User Dialog:** A dialog box for creating a new user "testuser" with "Enabled" checked. It also shows a preview of the user "admin" with credentials "admin" and "*****".
- Active Scan Panel:** Set up to start from <https://testfire.net> using the "Default Policy", context "https://testfire.net", user "testuser", and "Recurse" checked.
- Spider Panel:** Set up to start from <https://testfire.net> using the same context and user as the active scan.
- Alerts Panel:** Shows a list of 15 alerts found during the scan, including various security issues like Cross Site Scripting, SQL Injection, and Session Management problems.

Scan Results Analysis

Key Findings

Automated Scan

Detected multiple low and medium-risk vulnerabilities including missing security headers, outdated libraries, and potential XSS vectors.

Manual Exploration

Revealed hidden input fields, weak cookie configurations without HttpOnly/Secure flags, and exposed administrative endpoints.

Authenticated Testing

Discovered critical vulnerabilities behind authentication including SQL injection points, insecure direct object references, and privilege escalation paths.

Comprehensive reports summarized all findings categorized by severity (Critical, High, Medium, Low, Informational) with detailed remediation recommendations and OWASP references.

ZAP by Checkmarx Scanning Report

Generated with  ZAP on Tue 7 Oct 2025, at 02:49:32

ZAP Version: 2.16.1

ZAP by Checkmarx

Contents

- [About This Report](#)
 - [Report Parameters](#)
- [Summaries](#)
 - [Alert Counts by Risk and Confidence](#)
 - [Alert Counts by Site and Risk](#)
 - [Alert Counts by Alert Type](#)
- [Alerts](#)
 - [Risk=High, Confidence=High \(1\)](#)
 - [Risk=High, Confidence=Medium \(2\)](#)
 - [Risk=Medium, Confidence=High \(1\)](#)
 - [Risk=Medium, Confidence=Medium \(1\)](#)
 - [Risk=Medium, Confidence=Low \(1\)](#)
 - [Risk=Low, Confidence=High \(2\)](#)
 - [Risk=Low, Confidence=Medium \(4\)](#)
 - [Risk=Informational, Confidence=Medium \(3\)](#)
 - [Risk=Informational, Confidence=Low \(1\)](#)
- [Appendix](#)

Conclusion

Proven Capability

This project successfully demonstrated OWASP ZAP's comprehensive ability to detect, analyze, and report web application vulnerabilities across multiple scanning methodologies.

Security Best Practices

Emphasized the critical importance of continuous security scanning, regular vulnerability assessments, and timely remediation in maintaining application security.

Knowledge Growth

This hands-on project significantly strengthened practical understanding of web application security principles, vulnerability management workflows, and ethical penetration testing techniques.

THANK YOU