**National University of Computer and Emerging Sciences**

Information Security

**Project – Tool-Based (OWASP-ZAP)**

**Use-Case: SQL Injection vulnerability**

**NAMES: ROLL NUMBERS:**

Shoaib Asghar 19I-0406

Abdul Manan 19I-0500

Umar Mansoor 19I-0594

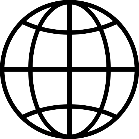
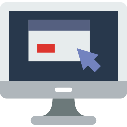
**DEGREE PROGRAM:** BS - CS

**SECTION:** CS-A

**COURSE Instructor:** Abdullah Abid

**DATE OF SUBMISSION:** December 5, 2022

**1. Introduction:**

OWASP ZAP (short for Zed Attack Proxy) is a web application security scanner. It is designed to be utilized by expert penetration testers as well as individuals who are new to application security. Its main purpose is to find the vulnerabilities in a web application. Under the auspices of the Open Web Application Security Project, the Zed Attack Proxy (ZAP) is a free, open-source penetration testing tool (OWASP). ZAP is versatile and adaptable and was created primarily for testing web applications. ZAP is fundamentally a "man-in-the-middle proxy," as the term is known. In order to intercept and analyze communications transmitted between the tester's browser and the web application, alter the contents as necessary, and then send those packets on to their intended recipient, it stands between the tester's browser and the web application. Both a standalone programmer and a daemon process may be utilized with it.

Browser

Web App

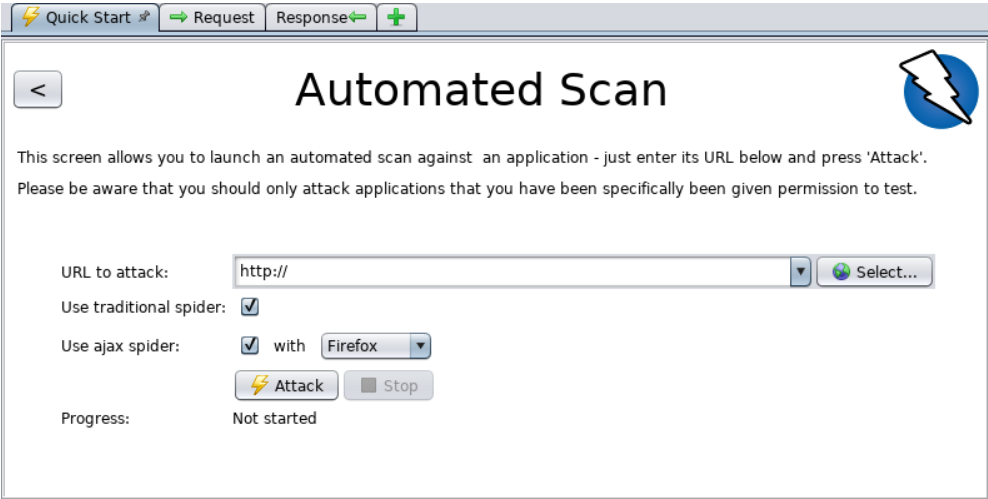
ZAP

Browser

**2. Internal Structure/Types of Scan in ZAP:**

**i). Automated Scan (Passive Scan):**

In Automated Scan, ZAP will continue to use its spider (Gather all POST and GET Requests of a Web app) to passively scan each page it discovers as it navigates around the web application. ZAP has two spiders for web application crawling; we may choose to utilize one or both of them from this screen. The conventional ZAP spider, which looks at the HTML in answers from the web application to find links. When examining an AJAX online application that creates links using JavaScript, this spider is quick, but it is not always successful. All of the requests and answers that are peroxided through ZAP will be passively scanned. ZAP has only performed passive scans of your web application thus far. Passive scanning is regarded as secure because it doesn't alter replies in any way. Additionally, scanning is done on a background thread to prevent lag.



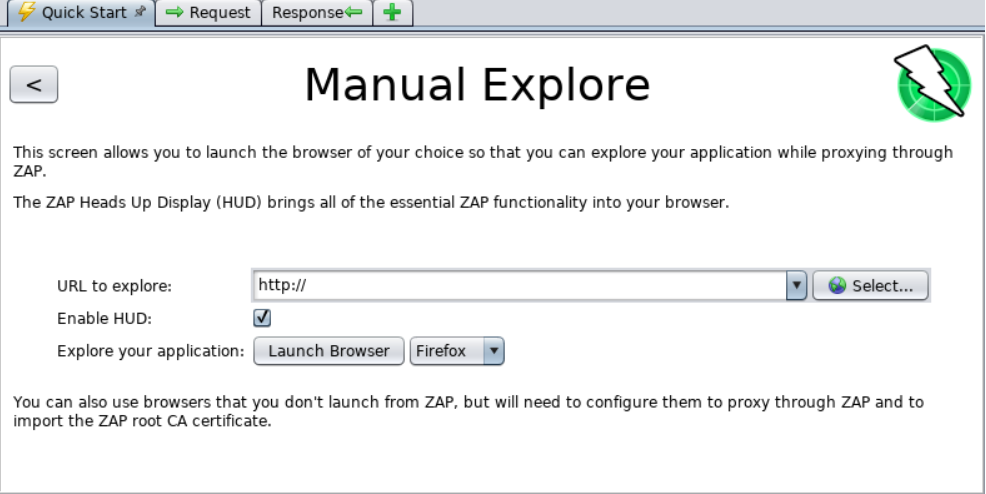
**ii). Manual Scan (Active Scan):**

Passive scanning and automated attack functionality is a wonderful place to start when assessing the vulnerabilities in your online application, but it has two flaws. And these are:

1. Pages protected by login pages are not discoverable during a passive scan since ZAP won't handle the necessary authentication unless its authentication feature is set.

2. In an automated assault, you have little control over the sorts of attacks that are launched or the order in which they are carried out. Outside of passive scanning, ZAP does provide a wide range of other choices for exploration and assaults.

Spiders are a fantastic tool for basic site investigation, but for maximum efficiency they should be used in co-existence with manual site exploration. Spiders, for instance, will only fill out the bare minimum of the default data in forms in your online site, whereas a user can submit more pertinent data, potentially exposing more of the web application to ZAP. This is particularly true for items like registration forms, which need a working email account. A random string entered by the spider might result in an error. When the form is submitted and accepted, more of the application may be revealed if a user responds.

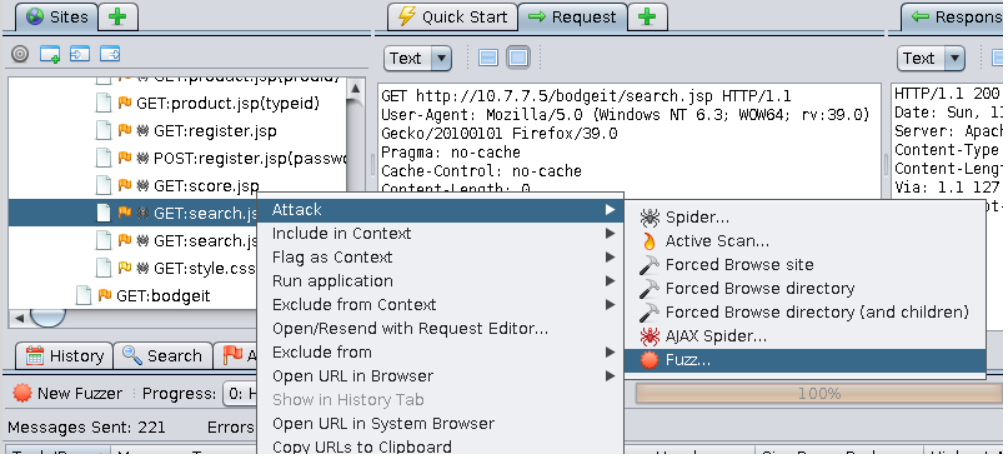


**3. Features:**

Although there are variety of features in ZAP, but we discuss here some of the most popular ones,

**i). ZAP Fuzzer:**

User may use a fuzzer with OWASP ZAP to provide a lot of unexpected or inaccurate data to an application under test. User have three options for payload creation: user’s own, any of the pre-built payloads, or payload add-ons made available by the ZAP community.



**ii). Web Sockets:**

Web Sockets will provide a genuine asynchronous communication channel between the client and the server, keeping the channel open and allowing the data to be sent in two directions (full-duplex). This is excellent for something like a chat programme. Vulnerability will also result from this since it keeps the channel open. ZAP will thus continue to scan Web sockets for vulnerabilities.

**iii). Spidering:**

When doing a penetration test, you use AJAX spidering to find requests on an AJAX-rich online application. A common spidering tool cannot detect these queries. By selecting Tools from the menu bar, then AJAX Spider, you may enter the AJAX spidering window (on). To assist you stop infinite crawling, this programme has configurable parameters like limit depth to crawl, maximum time, maximum crawl states, and other choices.

**iv). SQL Injection (Our main Use-Case):**

A SQL injection attack involves inserting, or injecting a SQL query through the client's input data into the programme. A successful SQL injection attack can access sensitive database data, alter database data (Insert/Update/Delete), and carry out database management tasks (such as shutdown the DBMS). A SQL query is "injected" into the application through the client's input data in a SQL injection attack. A successful SQL injection attack can modify database data (Insert/Update/Delete), access sensitive database information, and perform database administration operations (such as shutdown the DBMS). In ZAP, A security tester can perform advanced SQL Injection testing to determine whether the web application database is secure enough for SQL Injection. If certain request GET/POST is vulnerable to injection it will mark it as reflected and will generate an alert with that request. In addition to that it will also generate type of attack with the field i-e username, password etc with showing actually how attack can be made w.r.t sql Injection, For example by writing 1=1; or “Name’; etc. These types of input can generate sql Injection attack easily with that web app. So, ZAP can be very useful in that case, so that users can enhance their code for overcoming vulnerability.

