

OWASP ZAP TOOL

OWASP ZAP is a free, open-source web application security testing tool maintained by the Open Web Application Security Project (OWASP). It is designed to help security professionals and beginners identify vulnerabilities in web applications. ZAP works as an intercepting proxy, allowing testers to observe, modify, and analyze HTTP/HTTPS traffic between a browser and a web application.

At its core, ZAP acts as a man-in-the-middle proxy. It sits between your web browser and the target application, allowing it to intercept, inspect, and even modify the data being sent back and forth.

Key Features Include:

- ✓ **Automated Scanning:** Quickly identifies common flaws like SQL Injection and Cross-Site Scripting (XSS).
- ✓ **Passive Scanning:** Analyzes traffic in the background without modifying it (non-intrusive).
- ✓ **Active Scanning:** Actively attacks the application with known payloads to find deeper vulnerabilities.
- ✓ **Spidering:** Automatically "crawls" a website to map out its entire structure and hidden pages.
- ✓ **API & Automation:** Can be integrated into CI/CD pipelines to ensure security is checked with every code update.

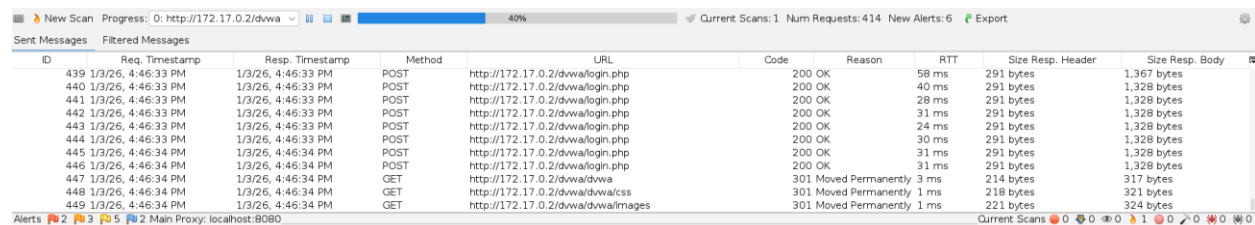
The screenshot displays the OWASP ZAP web application security tool interface. The main window is titled "Automated Scan" and contains a form for launching a scan. The "URL to attack:" field is populated with "http://172.17.0.2/dvwa". The "Use traditional spider:" checkbox is checked. The "Use ajax spider:" checkbox is unchecked, and the "with" dropdown is set to "Firefox Headless". The "Attack" button is highlighted. Below the form, the "Progress:" bar shows "Actively scanning (attacking) the URLs discovered by the spider(s)".

The bottom panel shows a table of scan results. The table has columns for ID, Req. Timestamp, Resp. Timestamp, Method, URL, Code, Reason, RTT, Size Resp. Header, and Size Resp. Body. The table contains 10 rows of data, showing various HTTP requests and responses, including POST requests to /dvwa/login.php and GET requests to /dvwa/dvwa and /dvwa/dvwa/css.

ID	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
439	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	58 ms	291 bytes	1,367 bytes
440	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	40 ms	291 bytes	1,328 bytes
441	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	28 ms	291 bytes	1,328 bytes
442	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	31 ms	291 bytes	1,328 bytes
443	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	24 ms	291 bytes	1,328 bytes
444	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	30 ms	291 bytes	1,328 bytes
445	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	31 ms	291 bytes	1,328 bytes
446	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	31 ms	291 bytes	1,328 bytes
447	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	GET	http://172.17.0.2/dvwa/dvwa	301	Moved Permanently	3 ms	214 bytes	317 bytes
448	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	GET	http://172.17.0.2/dvwa/dvwa/css	301	Moved Permanently	1 ms	218 bytes	321 bytes
449	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	GET	http://172.17.0.2/dvwa/dvwa/images	301	Moved Permanently	1 ms	221 bytes	324 bytes

The screenshot shows OWASP ZAP (Zed Attack Proxy) running an Automated Scan against a deliberately vulnerable web application, DVWA (Damn Vulnerable Web Application) hosted at

http://172.17.0.2/dvwa. The Automated Scan interface allows a tester to enter a target URL and launch a combined crawling and attack process. In this case, ZAP is actively scanning the application, as indicated by the progress bar (around 40%) and the status message showing that URLs discovered by the spider are being tested.



The screenshot shows the OWASP ZAP interface. At the top, a progress bar indicates the scan is at 40% completion. Below the progress bar, a table lists the HTTP messages sent during the scan. The table has columns for ID, Req. Timestamp, Resp. Timestamp, Method, URL, Code, Reason, RTT, Size Resp. Header, and Size Resp. Body. The messages include several POST requests to login.php and GET requests to dvwa, dvwa/css, and dvwa/images. The status bar at the bottom shows 2 alerts and 5 messages.

ID	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
439	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	58 ms	291 bytes	1,367 bytes
440	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	40 ms	291 bytes	1,328 bytes
441	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	28 ms	291 bytes	1,328 bytes
442	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	31 ms	291 bytes	1,328 bytes
443	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	24 ms	291 bytes	1,328 bytes
444	1/3/26, 4:46:33 PM	1/3/26, 4:46:33 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	30 ms	291 bytes	1,328 bytes
445	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	31 ms	291 bytes	1,328 bytes
446	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	POST	http://172.17.0.2/dvwa/login.php	200	OK	31 ms	291 bytes	1,328 bytes
447	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	GET	http://172.17.0.2/dvwa/dvwa	301	Moved Permanently	3 ms	214 bytes	317 bytes
448	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	GET	http://172.17.0.2/dvwa/dvwa/css	301	Moved Permanently	1 ms	218 bytes	321 bytes
449	1/3/26, 4:46:34 PM	1/3/26, 4:46:34 PM	GET	http://172.17.0.2/dvwa/dvwa/images	301	Moved Permanently	1 ms	221 bytes	324 bytes

At the bottom panel, the Active Scan tab displays real-time HTTP traffic generated during the scan. It shows multiple GET and POST requests sent to endpoints such as login.php, along with HTTP response codes like 200 OK and 301 Moved Permanently. This confirms that ZAP is interacting with the application by submitting forms and navigating pages to identify vulnerabilities. The counters indicate the number of requests sent and alerts found so far, demonstrating that ZAP is already detecting potential security issues.

CONCLUSION

ZAP provides both passive scanning, which detects issues without sending malicious requests, and active scanning, which safely attempts attacks to uncover vulnerabilities such as SQL injection, Cross-Site Scripting (XSS), authentication weaknesses, and security misconfigurations. Its Automated Scan feature combines crawling (spidering) and active testing, making it suitable for quick security assessments and learning environments. Overall, OWASP ZAP is a powerful tool for improving web application security and understanding common web vulnerabilities.