# Three Amigos Web Application Testing Tool Framework based on OWASP Top 10

Charles Corro
School of Computer Science
*University of Windsor*
Windsor, Ontario, Canada
corro@uwindsor.ca

Keerthana Madhavan
School of Computer Science
*University of Windsor*
Windsor, Ontario, Canada
madhava1@uwindsor.ca

Het Patel
School of Computer Science
*University of Windsor*
Windsor, Ontario, Canada
patel1ik@uwindsor.ca

Abstract— Web applications are used everywhere in today's world. They hold valuable information about people and many companies. Web applications can also perform great functions to help people with their daily needs like taxes, healthcare, etc. With the increase in data and usability within websites and web applications, Security is a big topic that needs to be addressed as more sensitive data is being held online. As such, developers and website creators must be ready to defend against any potential leaks of confidential data. Therefore, testing is an important step in securing the important data of both the developers and the users of their websites. This web application testing tool is meant to find, locate, and address the potential vulnerabilities of websites and web applications using the OWASP top 10 frameworks. Using two scans, one passive and one active, the web application tool will output a report for the user containing possible vulnerabilities along with resources to better understand and prevent an attack from it. This project hopes to highlight the importance of web application testing and the dangers of insecure websites, while also providing a better-streamlined method for developers to protect their websites.

Keywords—testing, web application, owasp, zap, flask

## I. Introduction

Web applications have evolved to cater to users of all kinds. Users benefit from websites in many ways. For example, Users can book flight tickets using a website making it a hassle-free task. User's data is stored by the company on their database which is very sensitive and must be kept confidential by the company. And as new features are added to the websites to handle complexities, the websites are prone to be vulnerable. Therefore, it is important for companies to make their websites secure by performing security testing and other testing methods that are favorable for their websites.

The statistics from Positive Technologies showed that 50% of the 38 fully functional websites had high-security issues [1]. Penetration testing is one testing method that is used to detect any potential security risks. Penetration testing involves manual or automated testing tools to test the websites. It is a simulation done by the "white hat hackers" which is intentional exploitation of systems to identify exploitable issues. These issues are then addressed by effective security controls. It can be performed by developers themselves.

## II. What is the Open Web Application Security Project (OWSAP)?

OWASP is an international organization that monitors overall web application security. OWASP 10 is a document that entails the ten most critical security risks of web applications. OWASP TOP 10 Web Application Security include [1]:

- A01:2021-Broken Access Control
- A02:2021-Cryptographic Failures
- A03:2021-Injection
- A04:2021-Insecure Design
- A05:2021-Security Misconfiguration
- A06:2021-Vulnerable and Outdated Components
- A07:2021-Identification and Authentication Failures
- A08:2021-Software and Data Integrity Failures
- A09:2021-Security Logging and Monitoring Failures
- A10:2021-Server-Side Request Forgery
- A02:2021-Cryptographic Failures
- A03:2021-Injection
- A04:2021-Insecure Design
- A05:2021-Security Misconfiguration
- A06:2021-Vulnerable and Outdated Components
- A07:2021-Identification and Authentication Failures
- A08:2021-Software and Data Integrity Failures
- A09:2021-Security Logging and Monitoring Failures
- A10:2021-Server-Side Request Forgery

However, these risks do not cover all the security aspects of a web application. Developers or the testing teams should therefore plan testing strategies according to the websites. Penetration testers use multiple tools to scan for vulnerabilities. A large number of tools used to scan can take up a lot of time [1]. Therefore, there arises a need for tools that can scan websites for multiple vulnerabilities at once. To keep up with the need our team has come up with a web tool that scans the website for vulnerabilities and displays a detailed report of the potential risks.

## III. Literature Review

For the purpose of the project, we did an overview of some research papers based on software testing. An SLR study suggested that the most dominant vulnerability is SQL injection followed by cross-site scripting and sensitive data exposure [2]. The code review that is in conjugation with the OWASP helps in verifying the security of the application's code. Two strategies mostly used in security testing are Black Box and White Box testing. Black box testing is performed by not referring to the logic and by feeding inputs to the system. The results are then checked to match the expected

results. Black box testing is also called Dynamic Analysis Security Testing (DAST). White box testing deals with the testing in reference to the internal code. It is also called Static Analysis Security Testing (SAST) [2].

The paper "Security testing of web applications" discusses various methods and approaches used in web application testing. It draws some important understanding for experienced and new developers to grasp guidelines for software testing. The case study of over 80 papers included the empirical results of the tools for software testing [2]. Another study identified 20 security techniques and categorized them into 3 levels: highest nodes, branching off into leaves in levels 2 and 3. A new taxonomy for better categorization of security techniques in the given 3 levels is also proposed [3]. PICO paradigm was used to identify methods for testing based on which three clusters were classified: security development models, lifecycle, and security tools and mechanisms. Some studies used security checks in the specification of their software. Their study suggested threat modeling was used by most researchers for security testing [4]. The researchers based on the finding from a study classified three different categories for the tools consisting of test management, functional, and load testing. The researchers based on the findings from a study classified three distinct categories for the tools consisting of test management, functional, and load testing. It reflects on the importance of automation tools that can help in identifying the hidden test cases and the need to keep up with the evolution of software and its associated risks [5]. The report on "Web Testing Platform Based on Hybrid Automated Testing Framework" discusses the need for a new approach-based testing framework that supports higher concurrency. The hybrid model that they proposed is based on keyword-driven testing that takes in tables and keywords to create the testing script. The authors' model is based on the analytical results of survivability of large-scale network and complex system testing [6].

## IV. Case Study

Test Case Three Amigos is a web-based Test Management tool used to test web application vulnerabilities and as well as display the results of running those tests. This tool is a new product that will be written in Python and Flask. The test team is responsible for testing the case study website Damn Vulnerable Web Application (DVWA) and ensuring it meets their needs. The test team is both the developer and the tester in this project.

Phase 1 of the project will deliver TCTA (Test Case Three Amigos) with functionality to test a website to identify its vulnerabilities or flaws based on the Open Web Application Security Project (OWASP) Principles. This will allow the test team to input a website URL and identify security flaws. Must have functionality is considered more important than the delivery date in this project because security problems are the root cause of damaged reputation and financial losses for most companies

## V. Case Study Scope

The initial phase will include all 'must have' requirements. These and any other requirements that are included must all be tested. At the end of Phase 1, a tester must be able to:

1. Input a "URL" of a website to we tested.
2. With a non-incremental approach, our tool will run and test for several cases.
3. Save it
4. Retrieve it and can then view the issue in detail
5. Mitigation Steps based on OWASP Recommendation
6. Rerun the scan

As the team works with the product and tests for flaws, it will be improved for accuracy. Load testing will not be considered part of this project since the user base is known and not an issue.

Our test environment is the google gruyere appspot in which we created an environment to test for vulnerabilities. Our environment id is 5264351517007722021185648214808648152 57.

## VI. Experimental Setup

In this project, Flask was used as the main framework of the web application. Flask is written in python and was the source of all the functionality within the server-side processes. Along with using Flask, the web application relied on various libraries such as JSON, zapv2, json2table, and others to compute results and communicate information from the front end to the back end. The main computational library was zapv2 as it allowed us to perform various tests and scans on websites with the URL code and API key. The front-end languages such as HTML, CSS, and JavaScript were used to display the information that was calculated on the back end.

Figure 5.0.1: flask logo

Figure 5.0.2: ZAP logo

## VII. Methodology

The main concept of this project was to scan for any vulnerabilities within a website using the respective URL and print a list of potential Vulnerabilities and some useful information. To start off we needed a website that was accessible and available to use as testing to see if our application could scan correctly. This site would need to purposefully have vulnerabilities for us to test and search for, without compromising anything such as getting blocked for

requesting data too many times or potential ethical sensitive data leaks that could be found. As such, google-gruyere was found. Google-gruyere is a site hosted by security experts that purposefully have multiple security bugs and vulnerabilities to help those learn more about web security and ways to stop attacks.

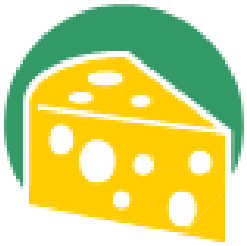

Figure 5.1.1: google-gruyere logo

After finding a suitable website to test our functions on, we used ZAP as our main library to scan and search the website for any vulnerabilities and security bugs. ZAP allowed us to scan each page of a website passively while using its spider. After a passive scan, an active scan is then started to attack the website's pages, functions, and tools. As such ZAP scans the website twice, first a quick passive scan, then a more in-depth active attack scan to find any issues that might occur based on OWASP standards. Once the scan is done, ZAP spits out the results of the list of vulnerabilities it found, along with the severity and potential workarounds to fix this vulnerability. Using ZAP as our main mediator between our web application and the website to the structure of the website, we are then able to compile the results and parse through to create a report which is shown to the user.


Figure 5.1.2: ZAP was used to scan and send reports

## VIII. About Three Amigos Testing Tool

The Tool was made for ease of use and simplicity. To use the tool, one must first download the repository along with all its dependencies at this link https://github.com/Three-Amigos-Testing/ThreeAmigoTest.git . After following the setup instructions within the README.md file, simply run the main app.py file by either double-clicking or typing the command "python app.py" in the command prompt terminal.

Once running simply copy the localhost URL into any browser. Afterward, you will be brought to the front page where you will simply input the website you wish to scan for vulnerabilities and click the submit button. Note the scan can take some time as it parses through every page of the website. As such, depending on the website, this may take a few minutes to complete. Please wait patiently on the loading screen, as you will be redirected once the scan has finished. Once finished you will be redirected to a page that contains a table of the vulnerabilities found and other useful information.

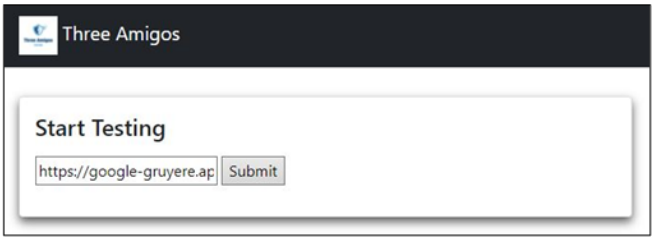


Figure 5.2.1: The first input prompt of the tool


Figure 5.2.2: The loading page while the scan is working

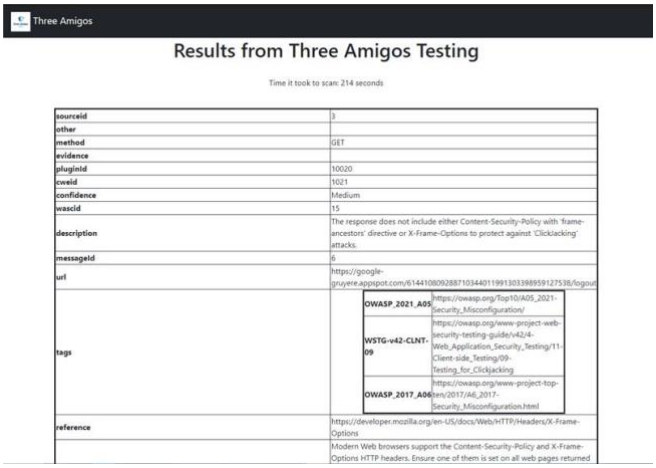


Figure 5.2.3: Sample of the final report page

## IX. Results and Discussion

As you see in the result above, our Three Amigos Scanner can detect top OWASP 10 vulnerabilities. In the result table output by our scan, you can see under the tag label, a link to the OWASP web application security testing vulnerability that contains further information on the vulnerability. Our scan also gives valuable insights to the developer on how we can fix such vulnerabilities on the solution label. Unlike [1] our tool runs a recursive test on each URL in the web directories, subdirectories, and even hidden directories to find vulnerable information.

## X. Future Work

Our Three Amigos web application security testing platform intends to provide users with an easier method to detect potential vulnerabilities that exist in a web application. There is no need to download software, purchase, or subscription to test vulnerable URLs and domains. The users do not require

to deal with the installation and usage of various web application penetration testing tools, our tools will provide all the necessary information from problem to potential solution. Finally, due to limited capabilities, this system has some weaknesses such as scan runtime and implementing better user interaction. However, it is possible to solve these problems in the future development increment process and hopefully, Three Amigos testing can be a sophisticated online penetration testing tool for users.

## REFERENCES


[1] J. Y. Zhong and M. M. Siraj, "Online Penetration Testing for Web Application Based on OWASP Framework." Universiti Teknologi Malaysia, Johor, Malaysia, 2021.

[2] Murat Aydos, Çiğdem Aldan,EvrenCoşkun, Alperen Soydan,
Security testing of web applications: A systematic mapping of the literature,

[3] O. B. Tauqeer, S. Jan, A. O. Khadidos, A. O. Khadidos, F. Q. Khan et al., "Analysis of security testing techniques," Intelligent Automation & Soft Computing, vol. 29, no.1, pp. 291–306, 2021.

[4] Musa Shuaibu, B., Md Norwawi, N., Selamat, M.H. et al. Systematic review of web application security development model. Artif Intell Rev 43, 259–276 (2015). https://doi.org/10.1007/s10462-012-9375-6

[5] K. Sneha and G. M. Malle, "Research on software testing techniques and software automation testing tools," 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), 2017, pp. 77-81, doi: 10.1109/ICECDS.2017.8389562.

[6] Z. Sun, Y. Zhang and Y. Yan, "A Web Testing Platform Based on Hybrid Automated Testing Framework," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), 2019, pp. 689-692, doi: 10.1109/IAEAC47372.2019.8997684.

[7] Z. A. Hamza and M. Hammad, "Web and mobile applications' testing using black and white box approaches," 2nd Smart Cities Symposium (SCS 2019), 2019, pp. 1-4, doi: 10.1049/cp.2019.0210.

[8] Lakshmi, D. & Mallika, S.. (2017). A Review on Web Application Testing and its Current Research Directions. International Journal of Electrical and Computer Engineering (IJECE). 7. 2132. 10.11591/ijece.v7i4.pp2132-2141.

[9] N. Singh, V. Meherhomji and B. R. Chandavarkar, "Automated versus Manual Approach of Web Application Penetration Testing," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2020, pp. 1-6, doi: 10.1109/ICCCNT49239.2020.9225385.

[10] M. Agreindra Helmiawan, E. Firmansyah, I. Fadil, Y. Sofivan, F. Mahardika and A. Guntara, "Analysis of Web Security Using Open Web Application Security Project 10," 2020 8th International Conference on Cyber and IT Service Management (CITSM), 2020, pp. 1-5, doi: 10.1109/CITSM50537.2020.9268856.