ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 14 (2024) No. 7, pp. 406-425
DOI:10.33168/JSMS.2024.0722

Application of the OWASP Framework to Identify and Remediate Vulnerabilities in Java Web Applications

Francisco Hilario, Diego Chang, Carla Zafra, Yesenia Vasquez, Laura Chipana

Universidad César Vallejo, Lima, Perú

fhilariof@ucvvirtual.edu.pe, ddelca@ucvvirtual.edu.pe, cmzafrar@gmail.com, yvasquez@ucv.edu.pe, lachipanar@ucvvirtual.edu.pe

Abstract. This study examines how the Open Web Application Security Project (OWASP) uses the methodology to assess security vulnerabilities in Java web applications. Through a literature review, a comparative evaluation of OWASP with alternative penetration testing methods was performed. Pre-experimental quantitative methodology was used for the analysis. The results of the audit of Java web applications conducted by 17 developers showed that the Open Web Application Security Project identified significant vulnerabilities, such as injection attacks, access control issues, and misconfiguration. The study demonstrates that OWASP performs well in assessing web application security threats and providing guidelines for correcting them. Its limitations include the lack of a comparison group and small sample size. Further research is needed on how to integrate OWASP with automation tools to make audits more efficient.

Keywords: Application security, Vulnerability Analysis, Java Web Application

1. Introduction

The Open Web Application Security Project methodology, which helps to have technical security controls in web applications, is not used in Java web applications due to a lack of knowledge and technical practice. Likewise, as a theoretical justification, technological aspects such as the practice of using the Open Web Application Security Project methodology were addressed, and the result and change are positive because it helps to measure the main security risks and threats to minimize them.

In addition, the methodology of the Open Web Application Security Project's techniques shows that there are several problems caused by incorrect programming within web applications. This is due to certain advances made by programmers, making insecure code vulnerable to attacks from questionable security and control systems. In addition, because hackers are performing higher level attacks on companies through new technologies coming out and previous ones are neglected, organizations today need to get more security in their programming. Continuing with the idea, Chavarria (2020) mentions that when performing a testing program, there is not always a right or wrong technical option to apply, and sometimes it can be difficult to choose between them. Although the ideal would be to perform tests in all possible areas, this is not always possible, and it may be necessary to perform penetration tests in several areas of the organization to obtain the best results. As a result, recent studies have shown that 75% of cyber-attacks target network security, compromising two-thirds of all online systems. Structured query language injection, parameter manipulation, directory traversal, cross-site request forgery (CSFR), denial of service (DoS), cookie poisoning, cross-site scripting (XSS), and session fixation are some of the attacks against these types of applications. National observations suggest that web applications should avoid these pitfalls because this problem is widespread. Web applications offer the advantage of being accessible from any web portal, regardless of the operating system, according to Romero (2019).

We point out the importance of implementing the OWASP methodology from a technological, social, and theoretical point of view. This has been stabilized by the Certified Java Application Security Engineer, which is used to detect flaws. Therefore, Gamboa (2021) pointed out that the multiple negative effects evidenced by vulnerabilities in security applications should be taken into account, as they represent a current threat to poor application development. As a result, this article explains how to implement and develop the Open Web Application Security Project methodology for web applications (Java) through computer security management. The planning and maintenance of the security programs of organizations deals mainly with strategic, tactical, and operational interviews (Bravo, 2019).

On the other hand, from the technological perspective of the study, it focuses on information on how to reduce threats or increase the level of security established for web applications that are registered in the Top Ten of OWASP with the most common security problems from the study of software and how to evade them, due to the lack of techniques and models that are detailing the efforts that exist to solve these problems (Menendez, 2022). The author Rio (2022) also described the weaknesses in the technological exploitation because it will start by performing the scanning and port recognition tool, which verifies the perimeter security controls, such as firewalls and IDS systems, and can protect the application, but not to the maximum since its controls are not effective enough to defend against attacks on the application layer.

According to Lala, Kumar and Subbulakshmi (2021), both large companies and individual developers are very concerned about ensuring the security of websites. During website development, it is more difficult to establish security measures the less common the technology used. Hackers look for unattended development and deployment vulnerabilities, which can cause large financial losses to companies or individuals. This not only affects developers but also exposes end users to vulnerable websites that could be vulnerable to XSS attacks, which could cause damage to their systems. Users who reuse passwords across multiple influential websites increase the risks. Every developer, regardless of experience, should consider addressing security gaps as a starting point for improving their

applications, as they are common in any development application. The goal of the project is to develop a secure web application that complies with the Open Web Application Security Project (OWASP) guidelines.

The growing popularity of web-based applications has complicated the protection of online data. The field of application security is constantly evolving due to advances in attack techniques, the introduction of new technologies with inherent vulnerabilities, the integration of comprehensive defense mechanisms, and the implementation of increasingly complex security systems. In this contribution, our goal is to develop a coherent model covering the most common and dangerous web attacks. This method will improve your understanding of these types of attacks and allow you to implement a security strategy that suits your particular technical and business circumstances (Ayachi et al., 2018). On the other hand, Damanik and Sunaringtyas (2020) demonstrated that the OWASP Test Guide is a tool for vulnerability testing because injection vulnerabilities are among the most important vulnerabilities identified. After testing, the code is reviewed using the OWASP Help Review Guide to find vulnerabilities in the source code. Finally, Damanik and Sunaringtyas (2020) provide security code suggestions that are based on the ideas used to address and mitigate vulnerabilities.

Thorough scanning and analysis of web applications is crucial to prevent or at least reduce potential security vulnerabilities. When applied to large code bases, traditional code review methods, such as manual review, present problems. Therefore, knowledge of automated code analysis tools is essential to evaluate their performance and reliability. The literature review found a variety of antecedents that facilitate the use of automated code analysis tools. Two static analysis tools were used in this case study to identify security threats in open-source web applications that had known vulnerabilities. The case study findings show that automated code review tools successfully identify security issues to an acceptable level and are significantly faster and more effective than manual reviews (Gholami, 2021). Therefore, evaluating source code for security issues has been a difficult undertaking. According to previous research, developers often overlook even obvious and easy-to-detect vulnerabilities during code review. Preliminary results indicate that this oversight may be related to reviewer bias and common practices. This study analyzes the effects of explicitly instructing developers to prioritize security during a code review and how it affects vulnerability detection. The study also analyzes the effect of providing a security checklist to guide the security review process (Braz, Çalıklı, and Bacchelli. 2022).

2. Literary Review

The segmented research shows the most relevant techniques on the OWASP Ten Best Practices methodology in Java web applications to improve academic training and secure code development in applications.

2.1. OWASP Methodology

Bergillos (2021) announced that the Open Web Application Security Project is a nonprofit institution to optimizes software security in web applications, that is, the Open Web Application Security Project foundation gives visibility, credibility, funding, and a broad community in open programs regarding security in the web platform, concluded with an awareness act for programmers indicating the most prominent security dangers in web applications.

2.2. Criteria of the methodology

Given the advances in attacks that occur against web applications, different criminal attacks can bring down a web application, which is why the OWASP methodology is used to help mitigate and correct through the use of secure code.

2.3. Security criteria

Chango and Gualpa (2019) released that they guarantee the security criteria of the methodology by

applying techniques from the Open Web Applications Security Project, managing sufficient risks in errors at the open source level to decree a regime of assistance of a computer security professional of contribution in knowledge, they concluded that this is intended to improve the security of web applications.

2.4. Vulnerabilities

Esquivel and Lozano (2020) indicated that the Open Web Application Security Project created 10 years ago, collected the main vulnerabilities, documented them, and admitted an end of risks by providing information on probabilities and impacts. It is currently one of the best and most used because every app and API developer is confident in its usefulness.

2.5. Penetration Testing

Nagendran (2019) stated that penetration testing and vulnerability assessment are different terms, in that it is appropriate to manifest the corresponding security flaws, while also including the exploitation of the discovered flaw and the attempt at data exfiltration. In addition, he indicated real-time testing of web applications that have been confirmed to be useful in defending the security of websites and concluded that web applications are mandatory after making the online request to avoid potential risks, so in this work, the techniques used to be tested are analyzed.

2.5.1. Tools for a test

Fernandez (2022) provided a list of tools that are being worked on but are not up-to-date and are not intended to augment previously created vulnerabilities. Some examples of devices: The open web application security project, LAPSE, is an open source tool created with Java security. The Open Web Application Security Project, or Orizon, is a source code program with a security scanner designed to detect vulnerabilities in web applications written in Java. OWASP Code Crawler, another tool developed by the Open Web Application Security Project for expert code review with static security code checks, is presented along with Insider, a free open source application intended for scanning stagnant web applications.

2.5.2. Openings toward a test

Freire (2022) proposed that the openings in information security systems determine the level of risk that I observe externally, and the level of occurrence, explaining that the probability of an attack on the assets found is through Open Source Intelligence.

2.6. Outreach Phase

Acosta (2022) presented that the scope testing phases consist of recasting the scanning of open ports with vulnerability analysis in which the information obtained is analyzed, and access is obtained for the post-exploitation phase, which makes the report of possible solutions, he began by making a list of network scans, Knowing that this stage is developed with vulnerability scanners, and infiltrates access to the system, he also described vulnerabilities linked to web applications, which explain misconfigurations or insecure practices such as Cross-site Scripting, and Structured Query Language Injection that has to filter the user fields entered by content.

2.7. Open Web Application Security Project License

Caucali (2020) noted that the purpose of his project was to conduct security testing on web applications to identify potential vulnerabilities caused by them, through penetration testing to diagnose and resolve security issues. By taking an open-source approach, we can identify and address the underlying reasons for the lack of security in software. In addition, it allows us to provide and verify security based on the specific characteristics of the application, making it easier to accurately design the requirements needed for information security.

2.8. Testing and Exploiting Tools to Improve Open Web Application Security Project

Web applications have experienced recent growth across various sectors such as public and private companies, government entities, and critical infrastructure. Consequently, ensuring the security of these web applications has become a significant priority to safeguard businesses against potential losses and unauthorized access to sensitive information. Developers may inadvertently introduce vulnerabilities by incorporating insecure modules or components from third parties, or by creating security flaws in their programming. Additionally, constraints such as tight budgets can contribute to these challenges. Unfortunately, these circumstances often lead to the oversight of a crucial security aspect in the lifecycle development of software (Alazmi. 2022).

2.9. Application Security Verification Standard (ASVS)

According to Blandón and Jaramillo (2023), OWASP, the Open Source Web Application Security Project, aims to identify and uncover the causes of software security breaches. OWASP establishes three levels of security assessment, increasing the comprehensiveness at each level. ASVS Level 1 is for any type of software; ASVS Level 2 is for applications that handle sensitive data and need protection; and ASVS Level 3 is for critical applications that perform high-value transactions, host sensitive information, or any application that requires the highest level of trust.

2.10. Software Assurance Maturity Model (SAMM)

Ahumada (2019), the open source web application security project delves into the open source framework that enables companies to develop and implement a software security strategy. Its adaptable implementation makes it suitable for companies of any size and development approach. Each of the twelve security practices defined in the model is composed of three maturity levels, which are grouped into four business functions related to software development.

2.11. Analysis of web security through the open web application security project

Willberg (2019) detailed that the open web application security project is a method within a security testing framework aimed at assessing the security of web applications to find vulnerabilities in a website. It aims to ensure that websites are protected through the use of checklists. The ten most dangerous categories of website vulnerabilities will be addressed through this initiative. These vulnerabilities include injection, faulty authentication, sensitive data exposure, extensible markup language external entities, compromised access control, security misconfigurations, cross-site scripting, untrusted deserialization, exploiting segments with known weaknesses, and other security-related issues.

2.12. Preventing attacks through the Open Web Application Security Project

Due to the remarkable technological advances that have occurred in recent years, people are increasingly adopting these emerging technologies. Although people rely heavily on these advances, there are several vulnerabilities in some fields that attract attackers who want to steal personal data. It is evident that cyber threats and breaches, caused by malicious motives or ransomware, are increasing globally. Users' servers and websites are frequently compromised, often without users being aware of the vulnerabilities. These vulnerabilities include the major issues identified by the Open Web Application Security Project, such as SQL injection and cross-site scripting. The suggested solution is to establish a web application firewall focused on the application layer of the Open Systems Interconnection (OSI) model to address and reduce these vulnerabilities.

2.13. Website vulnerability testing and web application scanning using Open Web Application Security Project

Priyawati, Rokhmah, and Utomo (2022) noted that many businesses, organizations, and social institutions use websites to facilitate their core tasks. Although websites have many benefits, it is essential to improve their security measures to reduce the risk of hacking. Cyber attacks or unauthorized access can have serious consequences, such as the compromise of important data. Therefore, assessing

the vulnerability of application features through comprehensive testing is crucial. Gray box penetration testing is a suitable testing method for distributed websites.

2.14. Technological tool

In addition, we can observe a comparison of methodologies in the implementation of web applications, where we observe the benefits and characteristics of the methodologies that are implemented in web applications. Table 1 explains the comparison of the different methodologies.

Table 1: Comparison of methodology in web application implementation

Indicators	Methodologies			
	OSSTMM	PTES	ISSAF	OWASP
Start Year	In 2000 the methodology was launched for the first time.	In 2009 the methodology was launched for the first time.	In 2006 the methodology was launched for the first time.	In 2003 the methodology was launched for the first time.
Versions	V4, v3, v2.1, v2.0	There are no versions	Draft 0.2.1, Draft 0.2	2021, 2017, 2016, 2013, 2010, 2007, 2004, 2003
Type Of Methodology	The complete procedure to perform penetration and reliability tests, security studies, and measurement of operational reliability to build the most robust security defenses for your company. (OSSTMM, 2010)	Number of information systems audited Characteristics of audited information systems The objectives of the audit Other organizational aspects. (PTES, 2014)	It is a structured, peer-reviewed framework that classifies information systems security assessments into areas and lists the evaluation or testing criteria for each of these areas. (ISSAF, 2017)	It is a security tool that presents a structure in which it increases exponentially, through processes used through detection tools and advice on measures to be taken to solve weaknesses. (OWASP, 2021)
Proceeds	It serves as evidence of a verification of tactics. Return to the examiner the responsibility for the test. Provide the client with a definitive result. Provide more complex details than an executive summary. Provide understandable metrics. (OSSTMM, 2010)	Have an accurate idea of the actual level of security of information systems and the maximum damage that a cyberattack can cause to the organization. provide data to assess the organization's cybersecurity compliance (PTES, 2014).	Aims to establish evaluation processes, against documents by evaluating the protections implemented against unauthorized access, finding technology-related misconfigurations, and strengthening the processes. (ISSAF, 2017)	It is a test that addresses security risks because it ensures avoiding the most common attacks and weaknesses through confidentiality, integrity, and availability through automated and manual tests that consider applications more robust and error-free. (OWASP, 2021)
Features	Environments are significantly more complicated compared to past seasons due to events such as virtualization, cloud computing, and remote operations. (OSSTMM, 2010)	In tests, it serves in a detailed way to carry out evaluation phases, which are tools that determine scope and limitations in an estimated time. (PTES, 2014)	Meets the requirements of evaluating an organization in the facet of security and assessment processes. (ISSAF, 2017)	It is considered to be an evaluation in web infrastructure, providing technological details with a broad and adequate vision achieving alignment with the standards from little to more effort. (OWASP, 2021)

Source: Authors.

3. Materials and Methods

After the analysis and review, it was determined that 6 documents were entered that were necessary to present the results of the study. Among the research, we cite studies because they met all the requirements. Table 2, details of the selected studies are presented:

Table 2: Cited documents linked to the size of the study.

N°	TITLE	AUTHOR	YEAR	COUNTRY
1	Analysis, development, and implementation of a security system to strengthen vulnerabilities and integrity of academic web applications. Thesis (Master's Degree in Telematics Security). Riobamba: Higher Polytechnic School of Chimborazo	Ramírez, F.	2022	Ecuador
2	Study of the main types of code injection attacks on web applications and systems to determine if a source code is vulnerable to STRUCTURED QUERY LANGUAGE Injection	Fernández, P.	2022	España
3	Computer Security, Methodologies, Standards and Management Framework in an Approach to Web Applications	Suarez, I. & Yagual, D.	2022	Ecuador
4	Design and implementation of a protection system against the injection attack Structured Query Language, on a vulnerable server using Open Access tools	Luque, A.	2021	Colombia
5	Diagnosis of computer vulnerabilities in the web applications of the Central University of Ecuador	Zambrano, G. & Andrade M.	2019	Ecuador
6	Use of Penetration Testing Technologies for Web Application Security Validation Based on OWASP's Top 10 Vulnerabilities	Zapata, J.	2018	Colombia

Source: Authors.

In reference to the collected studies it can be inferred that Ramirez (2022) conducted a study. According to the collected studies, it can be inferred that Ramirez (2022) created a comparative table of OSSTMM (Open Source for Security Testing Manual), PTES (Penetration Testing Execution), ISSAF (Information Systems Security Framework Assessment), and OWASP (Open Web Application Security Project) methodologies to evaluate the performance of techniques in web applications in Java. These methodologies develop penetration tests that compare similar criteria found in previous studies, and demonstrate that it is possible to exhibit, alter and/or expose breaches in Java web applications.

WebGoat, a web server platform with vulnerabilities through lessons on how to break and identify vulnerabilities, is one of the tools mentioned in the OWASP methodology. In addition, it has instructions on how to mitigate through the testing guide, as well as vulnerabilities for us to verify vulnerabilities and steps to test the ink tests in three black box cases. When the application is secure, a hacker is unknowingly looking for vulnerabilities. When you know or have information about the organization's framework or idea to look for vulnerabilities, the "gray box" is used. The white box provides information about penetration testing and how applications, source code, libraries, and vulnerabilities are built so that attackers cannot find vulnerabilities.

In addition, it is possible to identify the elective indicators or criteria of the current study. Among the most important elements are scope and approach; description of where testing is conducted and by whom; detailed activities related to the methodology; depth; detail of penetration and risk testing means used; usability; methodology layout; depth; between penetration testing detail and risk; and how to expose the implementation.

Table 3: Comparative table of methodologies in test penetration

Characteristics	OSSTMM	PTES	ISSAF	OWASP
Scope and Approach	Operational approach, for any company that intends to value information security.	Applicable with another methodology.	For all types of audits.	Targets web application media in any organization.
Scope	Monopolize computers and systems grouped into the network.	Develops and can successfully perform the test.	Average knowledge of details in pen-testing.	Audit web applications, in the full cycle of completion.
Depth	Specific analysis.	Security and penetration capabilities impact scanner performance.	Analyze the prerequisites before the evaluation.	Analyze the security of the application.
Usability	It requires training and is coded as an average limit in usability.	It is supported by other methodologies.	It is linear and protects the basic stages through the intrusion test.	Penetrating usability in web applications.
Metric	Determines the level of security of effectiveness, manipulates fair measurements called RAV (Risk Assessment Values, for its acronym in English).	Preserve information through previous studies.	It is re-evaluated by the same process to meet the evaluation criteria.	Has metrics to categorize and appreciate insecurities.
Risk estimation	Uses the RAV (Risk Assessment Values) considering accuracy, in the risk limits; and the equivalent substances in each module of operation.	Apply metrics by performing tests and proposing tools.	Applies countermeasures and recommendations through references and external documentation.	Apply metrics to assess and appreciate risks.

Source: Authors.

4. Results and Discussion

The web security manual, which identifies the business logic for security flaws detected by automated tools, served as the basis for the web auditing technique in this analysis. In addition, the application analysis finds the most frequent errors that significantly affect system security. In addition, the ninety controls described by the Open Web Application Security Project methodology were used to fully validate web security. As a result, it is recommended to complete an OWASP Top 10 web audit, as it allows the security of web applications in educational and commercial environments to be evaluated.

The Open Web Application Security Project methodology can be used in any project when applied to Java web applications. Table 4 provides detailed instructions on how to use OWASP TOP 10.

Table 4: OWASP TOP 10 analysis table in Java web applications

OWASP TOP 10	Utilities	Measures to be taken
A1: Injection	Incorrect attachment of input parameters.	• Strong passwords. • Database users and prefixes that are not by default. • A reliable security system housing.
A2: Loss of authentication	Guidelines for protecting credit card holder information during and after an online transaction.	• Strong passwords. • All versions must be original and secure. • Hide login errors. • Don't use more than two admins. • Check all again.

A3: Exposure to sensitive data	Security guidelines for safeguarding credit card holder data during and after an online transaction.	• A1 and A2 will be used. • The General Data Protection Regulation. • Hosting PCI Compliance and GDPR Compliance. • Controlling permissions for users. • Sensitive data is deleted. • HTTPS is synonymous with SSL.
A4: XML External Entity (XXE)	And external parties making incursions.	• Original and recognized software. • Upgrade SOAP to SOAP 1.2 or higher. • Although manual code review is the best alternative, SAST tools can help find XXE in source code.
A5: Broken Access Control	Access vulnerabilities are discovered using SAST and DAST tools.	• A1 and A2 will be used. • Examine JSON REST API calls for errors. • Save additional copies.
A6: Incorrect security settings	Data exposure and user protection.	• A1 and A2 will be used. • Use anything other than what's set as the default, including your computer, router, and phone.
A7: Cross-site scripting (XSS)	Browser security is at risk due to user sessions.	• Identify input vectors. • Analyze locations. • Test input vectors.
A8: Insecure decoding	Cause malicious code execution.	• Apply A1. • Use digital signatures as integrity checks for any serialized object.
A9: Components with vulnerabilities	Determining vulnerabilities requires extra effort.	• Avoid staying in unattended accommodations. • Avoid using software that has been removed or has known security flaws.
A10: Insufficient monitoring and logging	Offers numerous breaches and backdoors, some of which can be difficult to locate and repair.	• Obfuscated code. • Base code64. • File System Features. • GDPR (General Data Protection Regulation).

Source: Authors.

The web application security project testing guide focuses on web applications. As a result, we will use it as a guide to ensure reliable software development, which in turn will help us safeguard the pillars of IT security. However, to catalog the following tests/activities that actively involve information submission, it is essential to interact with both the applications and the servers that host them. When cybercriminals enter a computer center, they are not aware of what they are doing, so they manage to reach their target without verifying it, leading to the injection attack. Therefore, a flaw in the companies' login text fields is being investigated that could be exploited through a code leak known as injection, which aims to allow cybercriminals to enter the platform. These injection attacks target servers and applications. In addition, one of the most common attacks is against the database or operating system. This can be achieved through a web application, which could inject code and cause it to execute unexpectedly. The most common in the application are those that lack data entry validation; any type of input is potentially vulnerable; and there must be a restriction on the number of data that can be entered, so you should only allow database data to limit the number of characters.

On the other hand, measures are taken to reduce and correct injection attacks, such as application maintenance, security drivers, and database security patches. It is important that application developers and administrators are aware of the dangers associated with injections. In addition, during program development, special characters are filtered out and instructions are written incorrectly, which contributes to structured query language injection. In response, the client sends structured query language instructions through the publication and fetches variables that will be executed regularly. To specify, we should keep in mind that injection attacks are presented as threat agents and can be internal and external environment variables, web services, security weaknesses in legacy code found in Structured Query Language, No Structured Query Language, Lightweight Directory Access Protocol,

and Extensible Markup Language Path Language queries and that the effects of injection attacks cause a large amount of lost data as a result of data denial. Therefore, the methodologies mentioned in the tests were used to evaluate the main vulnerabilities shown in the Open Web Application Security Project Test Guide, along with the security tests evaluated in the deserialization and monitoring log of secure web applications. In addition, it is difficult to set up the tests integrated in the demo job with enormous certainty on web servers. The dimensions of the security mechanisms will determine these problems.

Before data processing, first of all, the normality test was applied using the Shapiro-Wilk statistical test because the sample was smaller than 50 participants. The significance value was $p < 0.05$; therefore, the data did not present a normal distribution. Consequently, nonparametric statistics were used.

$HE1_0$: Using the OWASP Ten Best Practices methodology will not increase optimization in web applications.

$HE1_1$: The use of OWASP's Top Ten Best Practices methodology increased optimization in web applications.

Table 5. Normality Test

	Shapiro-Wilk		
	Statistical	Gl	Gis.
Increased Optimization - Pre	,766	17	,001
Increased Optimization - Post	,262	17	,000

Pre-Test Optimization, Table 5, shows that the results after applying the normality test from the data measured in the pre-test optimization increase, the level was less than 0.05, which indicates that the sample does not fit the normal distribution.

Increased Post-Test Optimization, Table 5, shows that the results after applying the normality test from the data measured in the post-test optimization increase, the level was obtained to be less than 0.05, which indicates that the sample does not fit the non-normal distribution.

Table 6: Wilcoxon Signed Rank Test – Optimization Augmentation Towards OWASP Technical Methodology

Ranges				
		N	Average Range	Sum of Ranks
Increased Optimization - Pre - Increased Optimization - Post	Negative Ranges	9[a]	5,00	45,00
	Positive Ranges	0[b]	,00	,00
	Draws	41[c]		
	Total	17		
a. Increased Optimization - Pre < Increased Optimization - Post				
b. Increased Optimization - Pre > Increased Optimization - Post				
c. Increased Optimization - Pre = Increased Optimization - Post				

Table 6, shows the statistic on the optimization increase in the use of the OWASP technical methodology.

Table 7: Z-Test Statistic – Optimization Increase in OWASP Methodology

Test Statisticians	
	Increased Optimization - Post - Increased Optimization - Pre
Z	-3.448b
Asymptotic sig. (bilateral)	,001
a. Wilcoxon Sign Range Test	
b. It is based on negative ranges.	

After analyzing the data using the SPSS in the Z zone of Table 7, -3.448 was obtained, which was found in the rejection region and a value of $p = 0.001 < 0.05$ was obtained, therefore, the $HE1_0$ was

rejected, and the $HE1_1$: It was accepted that "The use of OWASP techniques methodology increased optimization in Java web applications".

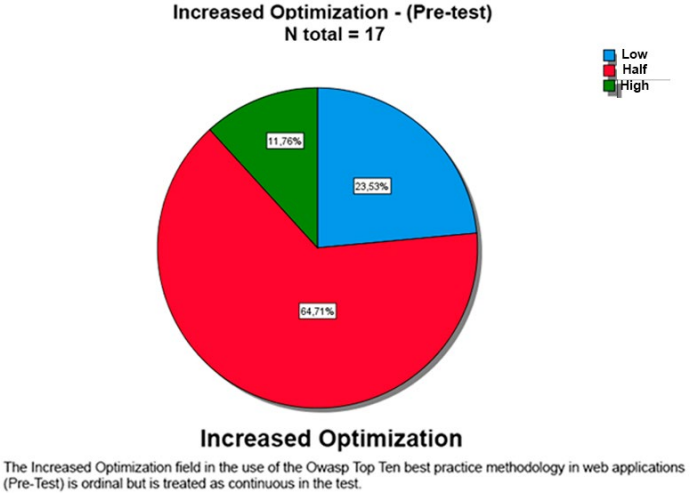

Fig 1: Optimization in the use of OWASP's Top Ten Best Practice methodology in web applications (Pre-Test)

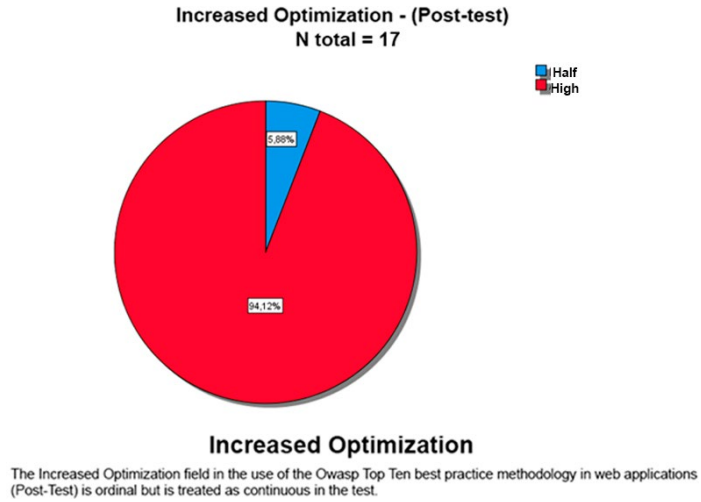

Fig 2: Increased optimization in the use of OWASP's Top Ten Best Practice methodology in web applications (Post-Test)

According to the results presented in the table, there is statistical evidence that supports the rejection of the null hypothesis, which allows us to conclude that the implementation of the OWASP techniques methodology for Java web applications significantly improves the increase in optimization. In addition, the figure shows an increase in the increase in optimization, going from a fair level to one considered as good.

$HE2_0$: Using OWASP's Ten Best Practices methodology will not lead to the growth of Java web applications.

$HE2_1$: The use of OWASP's Top Ten Best Practice methodology led to the growth of the Java web

Statistical data on the evolution of growth

For this indicator, an analysis was carried out with a group of 17 people who are dedicated to the development of web applications, by carrying out the OWASP technical methodology in Java web applications. The statistical tables according to the approach of the pre-test, and post-observation guide, where it was possible to measure the development of growth at the end of the OWASP technical methodology in Java web applications.

Table 8. Normality test

	Shapiro-Wilk		
	Statistical	Gl	Gis.
Growth Development - Pre	,802	17	,000
Growth Development - Post	,807	17	,000

Pre-Test Growth Development, Table 8 shows the results after applying the normality test from the data measured in the development of pre-test growth, it was obtained that the level is less than 0.05, which indicates that the sample does not fit the normal distribution.

Post-Test Growth Development, Table 8 shows the results after applying the normality test from the data measured in the development of post-test growth, it was obtained that the level is less than 0.05, which indicates that the sample does not fit the normal distribution.

Table 9: Wilcoxon Signed Rank Test – Growth Development Towards OWASP Technical Methodology

Ranges				
		N	Average Range	Sum of Ranks
Pre-Growth Development – Post-Growth Development	Negative Ranges	3a	2,00	6,00
	Positive Ranges	0b	,00	,00
	Draws	47C		
	Total	17		
a. Growth Development - Pre < Growth Development - Post				
b. Growth Development - Pre > Growth Development - Post				
c. Growth Development - Pre = Growth Development - Post				

Table 9, shows the statistics on growth development using the OWASP technical methodology.

Table 10: Z-Test Statistic – Growth Development in the OWASP Methodology

Test Statisticians	
	Pre-Growth Development - Post-Growth Development
Z	-1,732B
Asymptotic sig. (bilateral)	,083
a. Wilcoxon Sign Range Test	
b. It is based on positive ranges.	

After performing the analysis of the data using the SPSS in the Z zone of Table 10, -1.265 was obtained, which was found in the rejection region and a value of $p = 0.206 > 0.05$ was obtained, therefore, $HE2_0$ was not accepted and $HE2_1$ was accepted that "The use of OWASP's ten best practices methodology led to the growth of Java web applications".

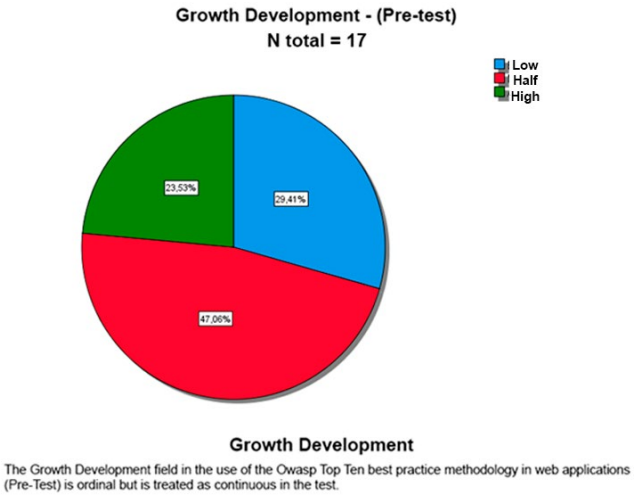

Fig 3: Development in the use of the OWASP Top Ten Best Practices methodology in web applications (Pre-Test)

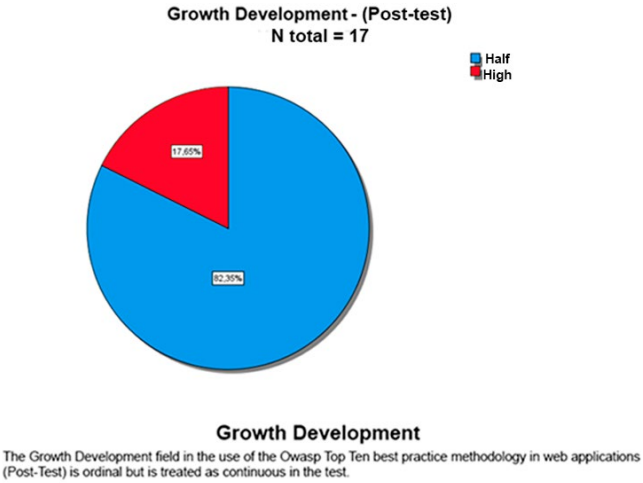

Fig 4: Development of growth in the use of the OWASP Top Ten Best Practices methodology in web applications (Post-Test)

According to the results presented in the table, there is statistical evidence that supports the rejection of the null hypothesis, which allows us to conclude that the implementation of the OWASP techniques methodology for Java web applications significantly improves the development of growth. In addition, the figure shows an increase in the development of growth, moving from a fair level to one considered good.

$HE3_0$: Using OWASP's Ten Best Practices methodology will not increase knowledge of web applications.

$HE3_1$: The use of the OWASP Ten Best Practices methodology increased knowledge of web applications.

Table 11. Normality test

	Shapiro-Wilk		
	Statistical	Gl	Gis.
Increasing Knowledge - Pre	,785	17	,001
Increasing Awareness - Post	,642	17	,000

Pre-Test Knowledge, Table 11, shows the results after applying the normality test from the data measured in the increase in pre-test knowledge, it was obtained that the level is less than 0.05, which indicates that the sample does not fit the normal distribution.

Increased Post-Test Knowledge, Table 11 shows the results after applying the normality test from the data measured in the increase in post-test knowledge, it was obtained that the level is less than 0.05, which indicates that the sample does not fit the normal distribution.

Table 12: Wilcoxon Signed Rank Test – Increasing Knowledge of OWASP Technical Methodology

Ranges				
		N	Average Range	Sum of Ranks
Increasing Knowledge - Post - Increasing Knowledge - Pre	Negative Ranges	7[a]	5,93	41,50
	Positive Ranges	3[b]	4,50	13,50
	Draws	7[c]		
	Total	17		
a. Increasing Knowledge - Post < Increasing Knowledge - Pre				
b. Increasing Knowledge - Post > Increasing Knowledge - Pre				
c. Increasing Knowledge - Post = Increasing Knowledge - Pre				

Table 12 shows the statistics on the increase in knowledge in the use of the OWASP technical methodology.

Table 13: Z-Test Statistic – Knowledge Increase in the OWASP Methodology

Test Statisticians	
	Increasing Knowledge - Post - Increasing Knowledge - Pre
Z	-1,513[b]
Asymptotic sig. (bilateral)	,130
a. Wilcoxon Sign Range Test	
b. It is based on negative ranges.	

After performing the data analysis using SPSS in the Z-zone of Table 13, -1.513 was obtained, which was in the rejection region and a value of $p = 0.130 > 0.05$ was obtained, therefore, $HE3_0$ was rejected, and $HE3_1$ was accepted that "The use of OWASP techniques methodology increased knowledge of Java web applications".

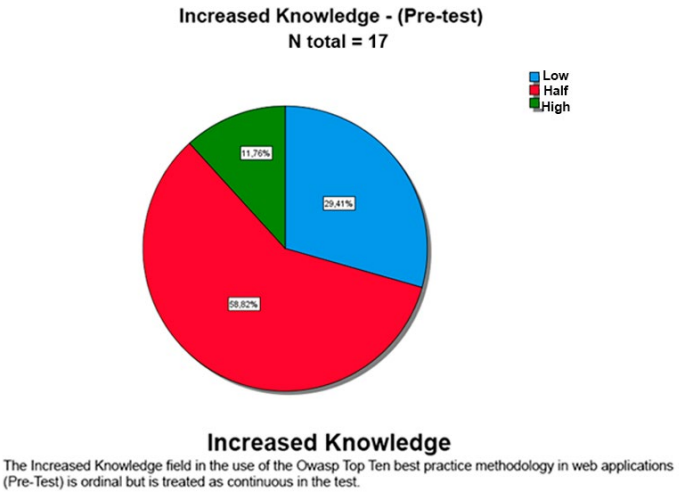

Fig 5: Knowledge of the use of the OWASP Top Ten best practice methodology in web applications (Pre-Test)

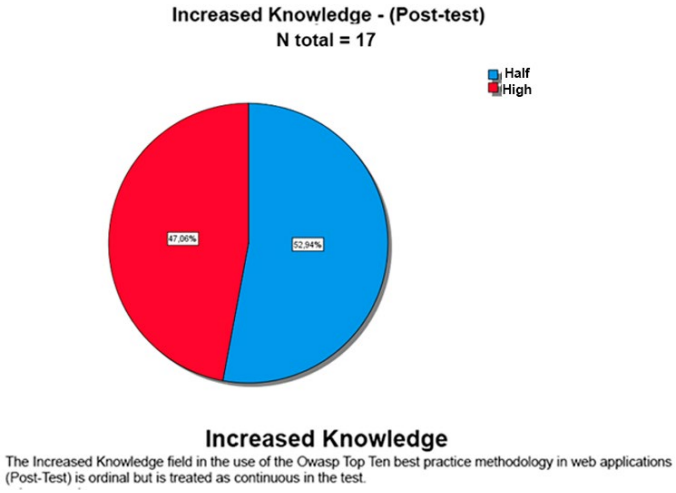

Fig 6: Increased knowledge in the use of the OWASP Top Ten best practice methodology in web applications (Post-Test)

According to the results presented, there is statistical evidence that supports the rejection of the null hypothesis, which allows us to conclude that the implementation of the OWASP techniques methodology for Java web applications significantly improved the increase in knowledge. In addition, the figure shows an increase in knowledge, moving from a fair level to one considered good.

$HE4_0$: The use of OWASP's Top Ten Best Practices methodology will not improve the level of security in Java web applications.

$HE4_1$: The use of OWASP's Top Ten Best Practices methodology improved the level of security in Java web applications.

Confirm that the data follows a normal distribution.

Table 14. Normality test

	Shapiro-Wilk		
	Statistical	Gl	Gis.
Security Level Enhancement - Pre	,815	17	,003
Improved Security Level - Post	,733	17	,000

Pre-Test Security Level, Table 14, shows the results after applying the normality test from the data measured in the improvement in the pre-test safety level, it was obtained that the level is less than 0.05, which indicates that the sample does not fit the normal distribution.

Improved Post-Test Security Level, Table 14, shows the results after applying the normality test from the data measured in the improvement in the post-test safety level, it was obtained that the level is less than 0.05, which indicates that the sample does not fit the normal distribution.

Table 15: Wilcoxon Signed Range Test – Improvement in the level of confidence towards the OWASP technical methodology

Ranges				
		N	Average Range	Sum of Ranks
Security Level Improvement - Post - Security Level Improvement - Pre	Negative Ranges	0a	,00	,00
	Positive Ranges	4b	2,50	10,00
	Draws	13C		
	Total	17		
a. Security Level Improvement - Post < Security Level Improvement - Pre				
b. Security Level Improvement - Post > Security Level Improvement - Pre				
c. Improvement in Security Level - Post = Improvement in Security Level - Pre				

Table 15 shows the statistics on the improvement in the level of security in the use of the OWASP technical methodology.

Table 16: Z-Test Statistic – Improvement in the Level of Security in the OWASP Methodology

Test Statisticians	
	Security Level Improvement - Post - Security Level Improvement - Pre
Z	-2,000b
Asymptotic sig. (bilateral)	,046
a. Wilcoxon Sign Range Test	
b. It is based on negative ranges.	

After analyzing the data using the SPSS in the Z zone of Table 16, -2.000 was obtained, which was found in the rejection region and a value of $p = 0.046 < 0.05$ was obtained, therefore, the $HE4_0$ was rejected and the $HE4_1$ was accepted, it was accepted that "The use of the OWASP best practices methodology improved the level of security in Java web applications".

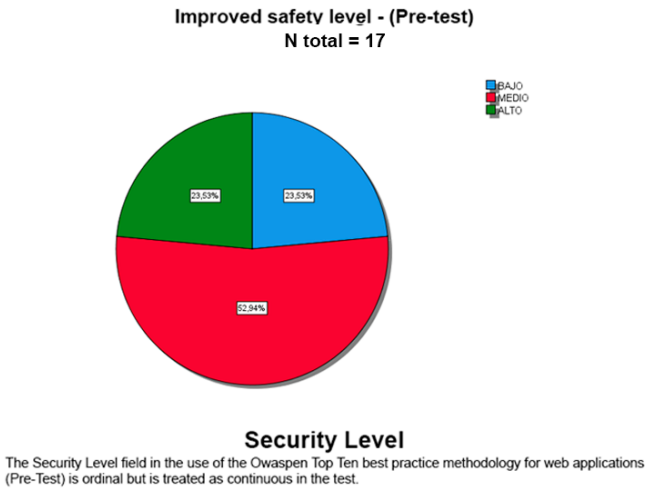

Fig 7: Level of Security in the use of the OWASP Top Ten Best Practice methodology in web applications (Pre-Test)

Fig 8: Level of Security in the use of the OWASP Top Ten Best Practice methodology in web applications (Post-Test)

According to the results presented, there is statistical evidence that supports the rejection of the null hypothesis, which allows us to conclude that the implementation of the OWASP techniques methodology for Java web applications significantly improves the level of security. In addition, there is an improvement in the level of security in the figure, going from a fair level to one considered good.

5. Conclusion

This study examined the security of Java web applications in comparison to other penetration testing techniques. The findings highlight OWASP's ability to assess vulnerabilities such as injection, broken access control, and misconfigurations. Important metrics such as accuracy, specificity, and risk assessment values demonstrate that OWASP's structured approach works. However, there are limitations, such as the amount of manual work required. To improve effectiveness, integration of OWASP with automated scanners should be considered in future work. In summary, this study demonstrates that the OWASP methodology is useful for developers to implement secure coding practices and for organizations to strengthen web application security.

Acknowledgments

We are very grateful to all the people who contributed knowledge and made this article a success.

References

Acosta Santana, J. J. (2022). Pentesting en entornos controlados [*Universidad de La Laguna*]. Disponible en:https://riull.ull.es/xmlui/bitstream/handle/915/28744/Pentesting%20en%20entornos%20controlados.pdf

Aljabri, M., Aldossary, M., Al-Homeed, N., Alhetelah, B., Althubiany, M., Alotaibi, O., & Alsaqer, S. (2022). Testing and exploiting tools to improve OWASP Top ten security vulnerabilities detection. *2022 14th International Conference on Computational Intelligence and Communication Networks (CICN)*. Disponible en: https://doi.org/10.1109/cicn56167.2022.10008360

Ahumada Munar, J. C. (2019). Análisis de metodologías para la implementación de un esquema de seguridad en el desarrollo de aplicaciones on line [*Universidad Nacional Abierta y a Distancia*]. Disponible en: https://repository.unad.edu.co/bitstream/handle/10596/33795/jcahumadam.pdf

Alazmi S, De Leon DC. A systematic literature review on the characteristics and effectiveness of web application vulnerability scanners. *IEEE Access*. 2022;10:33200-33219. doi:10.1109/access.2022.3161522

Ayachi, Y., Ettifouri, E. H., Berrich, J., & Bouchentouf, T. (2018). Modeling the OWASP most critical WEB attacks. En Smart innovation, systems, and technologies. Disponible en: https://doi.org/10.1007/978-3-030-03577-8_49

Bergillos Pedraza, S. (2021). La seguridad como punto de partida del desarrollo web [*Universidad de Girona*]. Disponible en: https://dugi-doc.udg.edu/bitstream/handle/10256/22561/Memoria_SergiBergillosPedraza_01-09-21.pdf

Blandón-Jaramillo, C. A., & Jaramillo-Becerra, J. S. (2023). Calidad del software y seguridad de aplicaciones a partir del proceso de desarrollo de software AGILISO y el estándar OWASP. *Revista Tecnología En Marcha*, 36(8), Pág. 5–22. https://doi.org/10.18845/tm.v36i8.6923

Bravo Mullo, S. J. (2019). Contribuciones para la Detección de Ataques Distribuidos de Denegación de Servicio (DDoS) en la Capa de Aplicación [*Universidad Mayor de San Marcos*]. Disponible en: https://cybertesis.unmsm.edu.pe/bitstream/handle/20.500.12672/18699/Bravo_ms.pdf

Braz, L., Aeberhard, C., Çalıklı, G., & Bacchelli, A. (2022). Less is more: supporting developers in vulnerability detection during code review. Proceedings of the 44th International Conference on Software Engineering. *ACM Digital Library*. Disponible en: https://doi.org/10.1145/3510003.3511560

Caucali Beltrán, D. M. (2020) Análisis y definición de requisitos de seguridad informática fundamentado en OWASP para el cumplimiento en los aplicativos basados en software libre en gestión documental. [*Universidad Nacional Abierta y a Distancia*]. Disponible en: https://repository.unad.edu.co/bitstream/handle/10596/38709/dmcaucalib.pdf

Chango Saavedra, R. J., & Gualpa Sarabia, D. A. (2023). Implementación de pruebas de hackeo ético para evaluar el sistema de seguridad informática en la empresa RHELEC Ingeniería CIA. LTDA [*Universidad Politécnica Salesiana*]. Disponible en: https://dspace.ups.edu.ec/bitstream/123456789/24450/1/TTS1228.pdf

Chavarría Gonzales, V. (2020). Estudio de los ataques contra website. OWASP [*Universidad de las Islas Baleares*]. Disponible en: https://dspace.uib.es/xmlui/bitstream/handle/11201/151259/Memoria_EPSU0643.pdf

Damanik, V. N. N., & Sunaringtyas, S. U. (2020). Secure Code Recommendation Based on Code Review Result Using OWASP Code Review Guide. *IEEE Xplore*. Disponible en: https://doi.org/10.1109/iwbis50925.2020.9255559

Esquivel Cabezas, H. A., & Lozano Olivares, J. H. (2020). Análisis de seguridad para el sitio web de la clínica veterinaria de occidente aplicando metodología de pentets owasp [*Universidad Nacional Abierta y a Distancia*]. Disponible en: https://repository.unad.edu.co/bitstream/handle/10596/36704/haesquivelc.pdf

Fernández Miranda, H. A. (2019). Análisis de la seguridad del sitio web del ministerio del trabajo aplicando pruebas de pentesting en la sede principal de la ciudad de bogotá [*Universidad Nacional Abierta y a Distancia*]. Disponible en: https://repository.unad.edu.co/bitstream/handle/10596/27059/hafernandezm.pdf

Freire Silva, J. E. (2022). Metodología para mitigar vulnerabilidades de almacenamiento mediante inteligencia de fuentes abiertas (OSINT) en la EEASA [*Pontificia Universidad Católica del Ecuador*]. Disponible en: https://repositorio.pucesa.edu.ec/bitstream/123456789/3528/1/77818.pdf

Gamboa Safla, D. L. (2021). Vulnerabilidades en aplicaciones web utilizando la metodología de "proyecto abierto de seguridad de aplicaciones web". Tesis de maestría, Ecuador [*Pontificia Universidad Católica del Ecuador*]. Disponible en: https://repositorio.pucesa.edu.ec/bitstream/123456789/3175/1/77336.pdf

Gholami, S. (2021). Automated Secure code review for web- applications. *DIVA*. Disponible en: https://www.diva-portal.org/smash/get/diva2:1587768/FULLTEXT01.pdf

Haq, I. U., & Khan, T. A. (2021). Penetration Frameworks and Development Issues in Secure Mobile Application Development: A Systematic Literature Review. *IEEE Access*, *9*, 87806–87825. https://doi.org/10.1109/access.2021.3088229

Helmiawan MA, Firmansyah E, Fadil I, Sofivan Y, Mahardika F, Guntara A. Analysis of Web Security using Open Web Application Security Project 10. 2020 8th International Conference on Cyber and IT Service Management (CITSM). octubre 2020. doi:10.1109/citsm50537.2020.9268856

Hilario, F., Milner Liendo, L. C., & Rivera, R. (2023). Evaluation of Cyber Sabotage in Public Entities. *Journal of System and Management Sciences*, *13*(4), 574-582. https://doi.org/10.33168/jsms.2023.0434

Hilario, F., Milner Liendo, L. C., C. Corpus, & Zafra, C. (2023). Evaluation of Algorithmic Metrics with A Focus on Server CyberRisks. *Journal of System and Management Sciences*, *13*(5), 322-338. http://dx.doi.org/10.33168/jsms.2023.0521

Higuera, J. R. B., Higuera, J. B., Sicilia, J. A., Riera, T. S., Argyros, C. I., & Magreñán, A. (2021). Combinatorial method with static analysis for source code security in web applications. *Cmes-computer Modeling in Engineering & Sciences*, *129*(2), 541-565. https://doi.org/10.32604/cmes.2021.017213

Hye-jin, K., & S., M. (2022). A Reinforcement Learning Model for Quantum Network Data Aggregation and Analysis. *Journal of System and Management Sciences*, *12(01), 11*. Disponible en: http://www.aasmr.org/jsms/Vol12/JSMS%20February%202022/Vol.12No.01.20.pdf

Ibarra-Fiallos, S., Higuera, J. B., Intriago-Pazmiño, M., Higuera, J. R. B., Sicilia, J. A., & Cubo, J. (2021). Effective filter for common injection attacks in online web applications. *IEEE Access*, *9*, 10378–10391. https://doi.org/10.1109/access.2021.3050566

Idris, M. Z., Syarif, I., & Winarno, I. (2021). Development of Vulnerable Web Application Based on OWASP API Security Risks. *IEEE ACCESS*. Disponible en: https://doi.org/10.1109/ies53407.2021.9593934

Ji-Yoon, K., & Chae-Kwan, L. (2022). The Use of Sentiment Analysis and Latent Dirichlet Allocation Topic-Modeling (LDA) on Web Novel Content Quality Fact. *Journal of System and Management Sciences, 12(2), 16*. Disponible en: http://www.aasmr.org/jsms/Vol12/JSMS%20April%202022/Vol.12No.02.11.pdf

Kennedy, M., Perkins, C., Brown, M., & Prins, K. (2022). Application security automation in development. 5(3), 216-226 (11). Disponible en: https://www.ingentaconnect.com/content/hsp/jcs/2022/00000005/00000003/art00004

Kiruba, B., Saravanan, V., Vasanth, T., & Yogeshwar, B. (2022). OWASP Attack Prevention. 2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC). *IEEE Xplore* Disponible en: https://doi.org/10.1109/icesc54411.2022.9885691

Kumar, S. A., & Rani, Y. U. (2022). Implementation and analysis of Web application security measures using OWASP Guidelines. *IEEE Access*. https://doi.org/10.1109/icmacc54824.2022.10093657

Lala, S. K., Kumar, A., & Subbulakshmi, T. (2021). Secure Web development using OWASP Guidelines. IEEE Access. Disponible en: https://doi.org/10.1109/iciccs51141.2021.9432179

Menéndez Arante, S. C. (2022). Auditoría de Seguridad Informática. *Grupo RA-MA*. https://www.digitaliapublishing.com/a/116389

Miao, L., Zhang, B., Chen, W., & Zhang, X. (2019). A survey of exploitation and Detection Methods of XSS vulnerabilities. *IEEE Access*, *7*, 182004–182016. https://doi.org/10.1109/access.2019.2960449

Mohammed, A., Alkhathami, J., Alsuwat, H., & Alsuwat, E. (2021). Security of Web Applications: Threats, Vulnerabilities, and Protection Methods. *International Journal of Computer Science and Network Security*, *21*(8), 167–176. https://doi.org/10.22937/IJCSNS.2021.21.8.22

Nagendran, K. (2019). Web Application Penetration Testing. In International Journal of Innovative Technology and Exploring Engineering (Vol. 8, Issue 10, pp. 1029–1035). Blue Eyes Intelligence Engineering and Sciences Engineering and Sciences Publication - *BEIESP*. Disponible en: https://doi.org/10.35940/ijitee.j9173.0881019

Priyawati, D., Rokhmah, S., & Utomo, I. C. (2022). Website Vulnerability testing and analysis of website applications using OWASP. *International Journal of Computer and Information System*, 3(3), 142-147. Disponible en: https://doi.org/10.29040/ijcis.v3i3.90

Río Hernández, O. (2022). Detección y explotación de vulnerabilidades del top 10 de OWASP y protección contra éstas [*Universidad Politécnica de Cataluña*]. Disponible en: https://upcommons.upc.edu/bitstream/handle/2117/375146/169364.pdf

Romero Aliaga, J. J. (2019). Trabajo de investigación de mejores prácticas en desarrollo de sistemas web contra ataque de inyección [*Universidad Tecnológica del Perú*]. Disponible en: https://repositorio.utp.edu.pe/bitstream/handle/20.500.12867/2078/Juan%20Romero_Trabajo%20de%20Investigacion_Maestria_2019.pdf

Shahid, J., Hameed, M. K., Javed, I. T., Qureshi, K. N., Ali, M., & Crespi, N. (2022). A Comparative Study of Web Application Security Parameters: Current trends and future directions. *Applied Sciences*, *12*(8), 4077. https://doi.org/10.3390/app12084077

Sierra Huertas, T. (2022). La seguridad informática en el desarrollo de aplicaciones web mediante el uso de la metodología OWASP [Universidad Nacional Abierta y a Distancia]. Disponible en: https://repository.unad.edu.co/bitstream/handle/10596/54049/ta52sie605.pdf

Sönmez, F. Ö. (2019). Security Qualitative Metrics for Open Web Application Security Project Compliance. *Procedia Computer Science*, *151*, 998–1003. https://doi.org/10.1016/j.procs.2019.04.140

Tudela, F. M., Higuera, J. B., Higuera, J. B., Sicilia, J. A., & Argyros, M. I. (2020). On combining static, dynamic, and interactive analysis security testing tools to improve OWASP Top ten security vulnerability detection in web applications. *Applied Sciences*, *10*(24), 9119. https://doi.org/10.3390/app10249119

Zambrano, K. B., Vidal, W. E. B., & Vera, R. R. T. (2022). Vulnerabilidades en los sistemas informáticos owasp top 10. *Journal Business Science, 3(2), 8*. Disponible en: https://revistas.uleam.edu.ec/index.php/business_science/article/view/221/308