International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V72I7P130

Volume 72 Issue 7, 278-283, July 2024

Original Article

Enhancing Security to Prevent Vulnerabilities in Web Applications

Shekhar Disawal[1], Ugrasen Suman[2]

[1,2] *School of Computer Science & IT, Devi Ahilya University, Indore, M.P., India.*

[1]*Corresponding Author : shekhar.disawal@gmail.com*

Received: 06 March 2024 Revised: 06 June 2024 Accepted: 14 June 2024 Published: 26 July 2024

Abstract *- The security of web applications remains a critical concern amidst escalating cyber threats and vulnerabilities. This research paper presents findings from an experimental study conducted on five websites using the pentest scanning tool. The experiment aimed to assess the vulnerabilities present in these web applications and identify potential security gaps. The prevalence of vulnerabilities such as SQL injection, Missing HttpOnly flag, and inadequate Content-Security-Policy underscores the urgent need for proactive measures to enhance web application security. Leveraging insights gained from the experiment, a novel Quality Enhancement Model for Secured Web Applications (QEMSWA) is proposed. This model integrates best practices and proactive strategies to fortify the security posture of web applications, addressing key areas such as the identification of assets, secure coding practices, code review, and effective vulnerability analysis. By proposing a recommendation model, this research seeks to empower organizations to mitigate risks and safeguard their web applications against emerging threats. Through the development of the QEMSWA model, this study contributes to ongoing efforts to establish a more resilient and secure digital environment.*

Keywords *- Web service, Web security, Vulnerability, Quality of security.*

1. Introduction

In today's digital age, the development and maintenance of secure web applications have become paramount due to the pervasive nature of cyber threats and the potential risks posed by data breaches. As organizations increasingly rely on web-based platforms to conduct business and interact with customers, ensuring the security and integrity of these applications is essential to safeguard sensitive information and maintain user trust. However, the prevalence of vulnerabilities in web applications, ranging from SQL injection to Cross-Site Scripting (XSS), highlights the urgent need for robust security measures to mitigate risks effectively. Recent years have witnessed a surge in cyber threats, with malicious actors exploiting vulnerabilities within web applications to perpetrate attacks with devastating consequences. One notable example is the OWASP Top Ten Web Application Security Risks report, which provides insights into the most prevalent vulnerabilities afflicting web applications. Analyzing data spanning from 2020 to 2023 reveals persistent challenges, including injection flaws, broken authentication, and sensitive data exposure, among others [2]. In response to these challenges, researchers and practitioners alike have devoted significant efforts to developing proactive strategies and tools aimed at bolstering the security posture of web applications. This research paper aims to contribute to this ongoing discourse by presenting novel insights and practical solutions to enhance the quality of security and prevent vulnerabilities in web applications. The paper begins by reviewing previous research in the domain of web application security, emphasizing the pervasive nature of vulnerabilities and the critical need for robust security measures throughout the Software Development Lifecycle (SDLC). It discusses the various approaches and methodologies employed by researchers to assess and mitigate security risks in web applications, including the evaluation of security scanners and the integration of security models into the SDLC [3]. Furthermore, the paper presents the results of an experimental study conducted on five distinct websites using open-source Pentest tools, identifying prevalent vulnerabilities and emphasizing the importance of proactive security measures. Leveraging insights gained from this study, the paper proposes a novel Quality Enhancement Model for Secured Web Applications (QEMSWA), which offers a structured approach to integrating security considerations throughout the web application development lifecycle. By empowering organizations with actionable recommendations and best practices, this research endeavors to enhance the resilience and security of web applications against emerging cyber threats. Ultimately, the goal is to foster a more secure digital environment for all stakeholders, mitigating risks and safeguarding sensitive information effectively. This paper is organized as follows: Section 2 presents a brief review of

related work. Section 3 provides an overview of web architecture. Section 4 describes the experimental setup and discusses the results. Section 5 introduces the Quality Enhancement Model for Secured Web Applications. Section 6 compares the proposed model with existing techniques. Finally, Section 7 provides concluding remarks.

2. Related Work

Enhancing the quality of security to mitigate vulnerabilities in web applications is paramount in today's digital landscape, where cyber threats loom large and data breaches pose significant risks to organizations and individuals alike. The prevalence of web-based attacks underscores the critical need for robust security measures to safeguard sensitive information and ensure the integrity of online systems. Previous research in this domain has highlighted the pervasive nature of vulnerabilities in web applications, ranging from SQL injection and Cross-Site Scripting (XSS) to authentication flaws and insecure configurations. Recognizing the escalating sophistication of cyber threats, researchers have endeavored to develop proactive strategies and tools to bolster the security posture of web applications. These efforts encompass a multifaceted approach, encompassing the identification, assessment, and remediation of vulnerabilities throughout the software development lifecycle. By synthesizing insights from prior studies and leveraging advancements in security technologies, this research aims to contribute novel insights and practical solutions to enhance the resilience of web applications against emerging threats and vulnerabilities.

Researchers concentrated solely on either commercial or open-source tools, while others conducted a thorough analysis covering both categories. A notable example is a recent and comprehensive comparison of security scanners aimed at organizations with limited resources, specifically small and medium-sized enterprises [4]. Another study conducted a thorough evaluation of ten web application assessment tools comprising a mix of both open-source and commercial scanners. The selected tools included Acunetix, AppScan, BurpSuite, Arachni, Pentest, NTOSpider, Paros, N-Stalker, Webinspect, and W3af to provide a comprehensive assessment of available options in the market. Their evaluation encompassed various criteria, such as vulnerability detection capabilities, ease of use, scalability, and accuracy of results.

By considering a diverse array of tools, the researchers aimed to offer insights into the strengths and weaknesses of each solution, aiding organizations in making informed decisions regarding their choice of web application security assessment tools [5]. This comprehensive evaluation approach ensured a holistic understanding of the capabilities and limitations of different tools, thereby contributing to the advancement of web application security practices. The development of web applications poses inherent risks due to the multitude of potential threats they encounter. Categorizing these risks stands out as a critical phase, as it determines the efficacy of integrating a security model into the Software Development Life Cycle (SDLC). If the categorization process is not thorough, it can lead to overlooking key vulnerabilities and undermine the effectiveness of the security measures implemented within the SDLC. Simply integrating a security model into the SDLC without addressing the identified security flaws adequately renders it ineffective [6,7]. Thus, ensuring a comprehensive risk categorization process is pivotal for optimizing the utility of security models integrated into the SDLC. The OWASP SAMM framework serves as a comprehensive resource for organizations seeking to enhance their software security strategy. It offers valuable tools and guidance, all available free of charge, to aid in several key areas [1].

The literature review highlights the critical importance of enhancing the security of web applications in response to the escalating cyber threats and vulnerabilities present in today's digital landscape. Researchers have extensively investigated various aspects of web application security, ranging from vulnerability assessment to risk categorization and integration of security measures into the Software Development Life Cycle (SDLC). Comprehensive evaluations of both commercial and open-source security tools have provided valuable insights into their strengths and limitations, aiding organizations in making informed decisions to mitigate security risks effectively. Moreover, frameworks such as OWASP SAMM offer practical guidance and resources for formulating robust software security strategies. Moving forward, continued research efforts and the adoption of proactive security measures are essential to ensure the resilience of web applications against evolving cyber threats, safeguarding sensitive information and maintaining the integrity of online systems.

Web applications are increasingly becoming targets for cyberattacks due to their ubiquity and the valuable data they often handle. Despite advances in security practices, vulnerabilities such as SQL injection, missing security headers, and improper session management continue to pose significant risks. These vulnerabilities can lead to data breaches, loss of user trust, and substantial financial losses. There is an urgent need for a comprehensive and proactive approach to identify and mitigate these security gaps effectively.

3. Overview of Web Architecture

Web applications consist of web pages and programs hosted on a web server, where user inputs are transmitted as parameter strings to generate SQL queries for data retrieval from a database. Authorized users access these applications over the internet or public networks to store and retrieve data, interacting through web browsers. Typically, web applications follow a three-tier architecture shown in Figure 1.

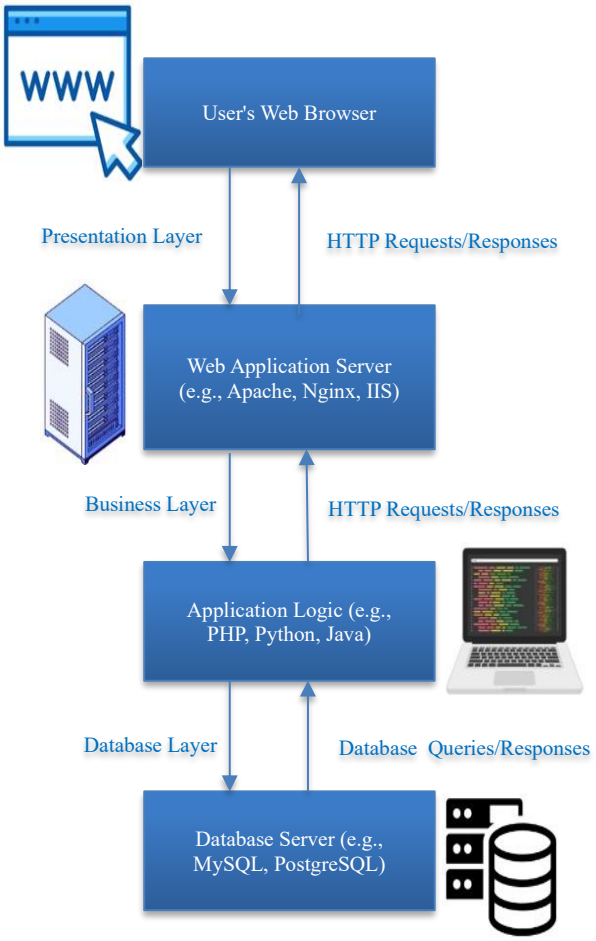

Fig. 1 The architecture of a typical web environment

3.1. Presentation Layer

This client-side layer processes information using CSS, HTML, and JavaScript.

3.2. Business Layer

This server-side layer includes code written in PHP, Java, Python, etc., responding to user requests.

3.3. Database Layer

Also, server-side, it manages data storage, retrieval, and provisioning according to user requirements.

4. Experimental Setup

The research conducted for this paper focused on utilizing open-source pentest tools to assess and identify vulnerabilities on five distinct websites, with the overarching goal of enhancing the quality of service and bolstering security measures to prevent potential risks in web applications. Open-source tools were chosen for their transparency, community-driven development, and accessibility, fostering a collaborative approach to cybersecurity. The results obtained were then categorized based on their severity and potential impact on the websites' security posture. The research aimed to democratize cybersecurity practices, allowing organizations with varying resources to access effective means of identifying and mitigating vulnerabilities. The findings contribute valuable insights into each website's specific risks and provide a foundation for implementing targeted security measures. Ultimately, this research seeks to contribute to the ongoing efforts to fortify web applications against potential threats, thereby enhancing the overall quality of service and user trust in online platforms.

4.1. Discussion

This discussion delves, according to Table 1, into the vulnerabilities discovered in various websites, emphasizing the significance of key security headers. One notable vulnerability is SQL Injection, which appears to be present in Website 3 and Website 5, both rated as 'High' severity. SQL Injection vulnerabilities can allow attackers to execute malicious SQL queries against the database, potentially leading to unauthorized access to sensitive data or even complete data loss. The presence of this vulnerability underscores the importance of robust input validation and parameterized queries to mitigate the risk of SQL Injection attacks. The X-Frame Options header, designed to safeguard against clickjacking attacks, is highlighted by the medium vulnerability observed in Website 2. This vulnerability raises concerns about potential exposure to clickjacking threats, emphasizing the necessity of implementing robust X-Frame Options across all websites to mitigate such risks. Moving on to the Referrer-Policy header, which influences the inclusion of information about the referring URL in HTTP headers, Website 2 stands out with a medium vulnerability, underscoring the need for comprehensive security measures to avoid potential leaks of sensitive information.

Table 1. Common vulnerabilities classified and detected by open-source scanners and their risk level on five distinct websites

Vulnerabilities	Website 1	Website 2	Website 3	Website 4	Website 5
SQL Injection	Not Found	Not Found	High	Not Found	High
Missing Security Header: X-Frame-Options	LOW	Medium	LOW	LOW	LOW
Missing Security Header: Referrer-Policy	LOW	Medium	LOW	LOW	LOW
Insecure Cookie Setting: Missing HttpOnly Flag	Not Found	Medium	Medium	Medium	Not Found
Insecure Cookie Setting: Missing Secure Flag	Not Found	Medium	Medium	Medium	Not Found
Missing Security Header: Content-Security-Policy	LOW	Medium	LOW	LOW	LOW
Missing Security Header: Strict-Transport-Security	LOW	Medium	LOW	LOW	LOW
Missing Security Header: X-Content-Type-Options	LOW	Medium	LOW	LOW	LOW

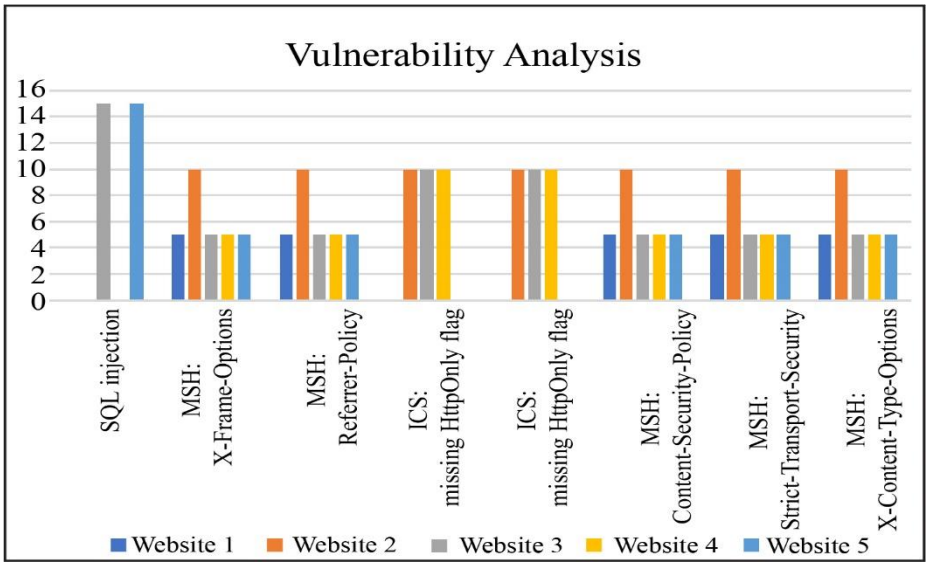

Fig. 2 Graphical representation of the web vulnerabilities

This discussion delves, according to Table 1, into the vulnerabilities discovered in various websites, emphasizing the significance of key security headers. One notable vulnerability is SQL Injection, which appears to be present in Website 3 and Website 5, both rated as 'High' severity. SQL Injection vulnerabilities can allow attackers to execute malicious SQL queries against the database, potentially leading to unauthorized access to sensitive data or even complete data loss. The presence of this vulnerability underscores the importance of robust input validation and parameterized queries to mitigate the risk of SQL Injection attacks. The X-Frame Options header, designed to safeguard against clickjacking attacks, is highlighted by the medium vulnerability observed in Website 2. This vulnerability raises concerns about potential exposure to clickjacking threats, emphasizing the necessity of implementing robust X-Frame Options across all websites to mitigate such risks. Moving on to the Referrer-Policy header, which influences the inclusion of information about the referring URL in HTTP headers, Website 2 stands out with a medium vulnerability, underscoring the need for comprehensive security measures to avoid potential leaks of sensitive information.

Addressing cookie security, the absence of both the HttpOnly and Secure flags in Websites 2, 3, and 4 is alarming, potentially exposing user data to attacks. This underscores the critical importance of configuring cookie security settings appropriately to ensure the protection of sensitive information. The Content-Security-Policy, vital for mitigating Cross-Site Scripting (XSS) attacks, reveals a medium vulnerability in Website 2, emphasizing the need for a robust CSP implementation to fortify defences against XSS threats and maintain the integrity of website content. The discussion also touches upon Strict Transport Security (HSTS), noting a medium vulnerability in Website 2. Implementing HSTS is deemed fundamental for preventing man-in-the-middle attacks and enhancing overall security, warranting immediate attention. Lastly, the X-Content-Type-Options header, crucial for reducing the risk of Multipurpose Internet Mail Extensions (MIME) sniffing attacks, showcases mostly low vulnerabilities across websites. However, maintaining a consistently low vulnerability level across all websites is deemed critical for ensuring a robust overall security posture. Examining the vulnerabilities across different websites underscores the need for a comprehensive approach to web security. For this we have proposed a quality enhancement model for web applications which gives a guideline for website administrators to prioritize the implementation of essential security headers, consistently maintaining low vulnerabilities across all aspects. This model will regularly provide security audits and updates which are essential to stay ahead of evolving cyber threats and safeguard user data and online experiences.

5. Quality Enhancement Model for Secured Web Applications

We have presented a Quality Enhancement Model for Secured Web Applications (QEMSWA) framework that systematically addresses various aspects of web application security throughout its lifecycle. Figure 3 presents the Quality Enhancement Model for Secured Web Applications. The model has five different phases: i.e. Assessment Phase, Requirement Analysis, Development Practices, Testing Strategies, and Continuous Monitoring. The assessment phase is the initial step, involving the identification of assets, threat modelling, and vulnerability analysis. Following this, the requirement analysis phase focuses on understanding the security needs of the application, defining security requirements, and aligning them with business objectives. The development practice stage emphasizes secure coding practices, secure architecture, and robust implementation of security controls. Testing strategies are then employed,

including static and dynamic analysis, penetration testing, and code reviews, to validate the effectiveness of security measures. The continuous monitoring phase involves real-time surveillance of the web application's security posture, ensuring proactive detection and response to emerging threats. This model integrates security considerations seamlessly into each stage of the web application development process, providing a structured approach to enhance the quality and security of web applications. Through a holistic and iterative approach, this model contributes to a more resilient and trustworthy web application ecosystem, aligning with the evolving landscape of cybersecurity challenges.

A detailed description of each phase is as follows:

5.1. Assessment Phase

- Identification of Assets: Enumerate and classify all assets involved in the web application, including data, infrastructure, and components.
- Threat Modelling: Systematically analyse potential threats, attack vectors, and security weaknesses through a structured approach, identifying potential risks.
- Vulnerability Analysis: Conduct a thorough examination of the web application's codebase and infrastructure to identify and assess vulnerabilities.

5.2. Requirement Analysis Phase

- Security Needs Analysis: Collaborate with stakeholders to understand the specific security needs and concerns of the application, considering regulatory requirements and industry standards.
- Security Requirements Definition: Clearly articulate security requirements, specifying the necessary controls, encryption standards, access controls, and authentication mechanisms.
- Alignment with Business Objectives: Ensure that security requirements align with the overall business objectives of the web application to create a balance between security and functionality.

5.3. Development Practice Stage

- Secure Coding Practices: Enforce coding standards that prioritize security, emphasizing secure coding practices and techniques to prevent common vulnerabilities.
- Secure Architecture: Design a secure architecture incorporating principles like the principle of least privilege, defence-in-depth, and proper data flow controls.
- Implementation of Security Controls: Actively integrate security controls into the development process, utilizing frameworks and libraries that enforce secure practices.

5.4. Testing Strategies

- Static Analysis: Employ static analysis tools to review the source code for vulnerabilities, ensuring early detection of potential security issues.
- Dynamic Analysis: Conduct dynamic testing through methods like automated scanning and simulated attacks to assess the web application's behaviour under various conditions.
- Penetration Testing: Engage in ethical hacking activities to identify and exploit vulnerabilities that may not be apparent through automated tools.
- Code Reviews: Regularly review codebase for security issues, involving peers in the validation process to ensure a comprehensive assessment.

5.5. Continuous Monitoring Phase

- Real-time Surveillance: Implement continuous monitoring tools and processes to observe the web application's security posture in real-time actively.
- Proactive Detection: Employ intrusion detection systems, log analysis, and anomaly detection to identify and respond to potential security incidents proactively.
- Response to Emerging Threats: Develop and implement incident response plans that enable swift and effective responses to emerging threats, minimizing potential damage.

6. Comparison of the Proposed Model with the Existing Techniques

The findings of this analysis show a significant occurrence of vulnerabilities throughout the examined sites. Among the reported vulnerabilities are SQL injection, a missing HttpOnly flag, and an inadequate Content-Security Policy. These findings are consistent with the OWASP Top Ten Web Application Security Risks, which highlight the ongoing issues in web application security.

Fig. 3 Quality Enhancement Model for Secured Web Applications (QEMSWA)

Several major elements contributed to the study's superior findings when compared to cutting-edge procedures mentioned in the literature:

- The Pentest scanning tool enabled a thorough assessment of each website, revealing a greater range of vulnerabilities than is generally documented in the literature. This detailed examination offered a better knowledge of the security flaws found in web apps.
- The research emphasized the use of open-source tools, which offer transparency and community-driven development. This approach enabled a more collaborative and adaptable security assessment, leveraging the latest updates and improvements from the cybersecurity community.
- The proposed QEMSWA model integrates best practices from the literature and the latest industry standards. By incorporating secure coding practices, regular code reviews, and continuous vulnerability analysis, the model ensures a proactive and holistic approach to web application security.
- The QEMSWA model is designed to be iterative, allowing for continuous monitoring and improvement of security measures. This iterative approach ensures that web applications can adapt to emerging threats and evolving security requirements, maintaining a robust security posture over time.

7. Conclusion

The research paper has demonstrated the critical importance of addressing vulnerabilities in web applications amidst escalating cyber threats. Through an experimental study conducted on five distinct websites using a Pentest scanning tool, prevalent vulnerabilities such as SQL injection, X-Frame-Options, Referrer-Policy, Missing HttpOnly flag, and Content-Security-Policy were identified, highlighting the urgent need for proactive security measures.

Leveraging insights gained from this study, a novel Quality Enhancement Model for Secured Web Applications (QEMSWA) is proposed. This phased approach ensures a thorough and systematic integration of security considerations throughout the web application development lifecycle, promoting a resilient and trustworthy application ecosystem.

Each phase builds upon the previous one, creating a holistic model that adapts to the evolving landscape of cybersecurity challenges. By empowering organizations with a comprehensive recommendation model, this research endeavours to enable them to effectively mitigate risks and safeguard their web applications against emerging threats. Implementing the QEMSWA model is essential to test its significant contribution to fostering a more resilient and secure digital environment for all stakeholders.

References

[1] Software Assurance Maturity Model - A Guide to Building Security into Software Development - Version 1.0, OWASP, pp. 1-96, 2010. [Online]. Available: https://opensamm.org/downloads/SAMM-1.0.pdf

[2] Top 10 Web Application Security Risks, OWASP. [Online]. Available: https://owasp.org/www-project-top-ten/

[3] Gergely Trifonov, "Reducing the Number of Security Vulnerabilities in Web Applications by Improving Software Quality," *2009 5th International Symposium on Applied Computational Intelligence and Informatics*, Timisoara, Romania, pp. 511-54, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[4] Ricardo Araújo, António Pinto, and Pedro Pinto, "A Performance Assessment of Free-to-Use Vulnerability Scanners - Revisited," *ICT Systems Security and Privacy Protection, IFIP Advances in Information and Communication Technology*, Oslo, Norway, vol. 625, pp. 53-65, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Adam Doupé, Marco Cova, and Giovanni Vigna, "Why Johnny Can't Pentest: An Analysis of Black-Box Web Vulnerability Scanners," *Detection of Intrusions and Malware, and Vulnerability Assessment: 7th International Conference*, Bonn, Germany, pp. 111-131, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[6] Sushila Madan, and Supriya Madan, "Security Standards Perspective to Fortify Web Database Applications from Code Injection Attacks," *2010 International Conference on Intelligent Systems, Modelling and Simulation*, Liverpool, UK, pp. 226-230, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[7] C. Striletchi, and M.F. Vaida, "Enhancing the Security of Web Applications," *Proceedings of the 25th International Conference on Information Technology Interfaces*, Cavtat, Croatia, pp. 463-468, 2003. [CrossRef] [Google Scholar] [Publisher Link]