Journal of Informatics and Vocational Education (JOIVE)
Vol.7, No.3, November 2024, pp. 114~123
ISSN: 2746-7813

Security Analysis of Web-based Information Systems Through Vulnerability Assessment Using the Framework of OWASP Web Security Testing Guide and Common Vulnerability Scoring System

Muhammad Rifqy Abdallah, Puspanda Hatta*, Cucuk Wawan Budiyanto
Department of Informatics Education, Universitas Sebelas Maret, Indonesia

Article Info

Article history:

Received Jun 05, 2024
Revised Sep 25, 2024
Accepted Sep 30, 2024

Corresponding Author:

Puspanda Hatta,
Department of Informatics Education, Universitas Sebelas Maret, Jl Ahmad Yani, Pabelan, Kartasura, Surakarta, Central Java, 57169, Indonesia.
Email:
hatta.puspanda@staff.uns.ac.id

ABSTRACT

The technology of web-based information systems continues to develop and has been adopted by many organizations, including higher education institutes. However, this technology carries inherent security risks, making regular security analysis essential. This research presents a case study of eight web-based information systems at a higher education institution to assess the security condition of each system individually and its overall characteristics, as well as to construct an advanced strategy for maintaining and optimizing system security. The security analysis was conducted using a mixed-method approach: qualitatively through the OWASP Web Security Testing Guide framework across four categories (Information Gathering, Configuration and Deployment Management Testing, Session Management Testing, and Client-side Testing); and quantitatively through Common Vulnerability Scoring System (CVSS) calculations. All information systems tested were found to be vulnerable, though with varying levels of severity. Vulnerability discovery ratios ranged from a low of 8% (with a 'Low' severity level) to a high of 31% (with severity levels reaching 'Critical'). Overall, systems based on a Content Management System (CMS) were found to be less vulnerable compared to those built on non-CMS frameworks. Based on the discovered vulnerabilities, follow-up recommendations were constructed to serve as a reference for improving and optimizing the systems' security.

Keywords: Common vulnerability scoring system, information system, owasp web security testing guide, vulnerability assessment, web security

1. INTRODUCTION

Various organizations have been adopting web-based information systems to manage their business activities (Mateus & Serrão, 2021). This technology is considered a solution that brings convenience (Aziz, 2021) and can lead to improvements in the productivity, excellence, and quality of organizations (Kusuma, 2022).

Besides carrying convenience, web-based information systems also carry a number of security risks (Andria, 2020). Several cyberattacks can be launched against a system, such as Cross-Site Scripting, Denial of Service, SQL Injection, and many others (Sahren et al., 2019). A security breach could lead to sensitive data being stolen, financial loss, a decline in the company's reputation, or even a complete loss of control over the system (Ravindran & Potukuchi, 2022).

Given these risks, it is important to perform a security analysis on the system to protect it from the various threats of cyberattacks that could happen, including vulnerability attacks (Subandi & Sugara, 2022). A vulnerability attack is a cyberattack conducted by exploiting security holes within the system (Putra & Soewito,

2022). These holes can take the form of an insecure system architecture, misconfigurations in various components, the use of outdated and vulnerable components, and more (OWASP Foundation, 2021).

One step that can be taken to secure a system from such vulnerability attacks is through vulnerability assessment (Tania et al., 2018). As a security audit activity, vulnerability assessment can be conducted by following security audit guides or frameworks, such as the OWASP Web Security Testing Guide, commonly known as OWASP WSTG (Rafeli et al., 2022). By following a framework, the audit can be performed using sequential processes. The discovered security holes will have consistent, accurate, and reproducible results, and will be well-documented so they can be handled properly and thoroughly (OWASP, 2020).

A literature review on open journal repositories found that many web security tests are conducted without following any frameworks and/or only report the vulnerabilities found without any follow-up plan to address them (Andria, 2020; Awlarijal et al., 2020; Aziz, 2021; Ghozali et al., 2019; Kusuma, 2022; Listartha et al., 2022; Mateus & Serrão, 2021; Mu'min et al., 2022; Rosaliah et al., 2021; Sahren et al., 2019). It was also found that research using the OWASP WSTG framework has not been widely conducted. Although the WSTG was released in 2020, much research still uses the older version of the OWASP guide, namely the OWASP Testing Guide (OTG), which is the predecessor of OWASP WSTG (Anggraeni et al., 2022; Daniswara et al., 2020; Guntoro et al., 2020; Subana et al., 2020). Various studies also perform security tests on campus websites, but their research object is limited to a single website (Anggraeni et al., 2022; Aziz, 2021; Daniswara et al., 2020; Guntoro et al., 2020; Kusuma, 2022; Maharani et al., 2017; Mu'min et al., 2022; Subana et al., 2020; Yudiana et al., 2021).

Based on the literature review, no research was found that conducted a comprehensive vulnerability assessment on multiple web-based information systems at once using the OWASP WSTG framework—a study that covers the entire process from the testing stage and creation of security metrics to the construction of a follow-up strategy. Therefore, there is a gap in the literature for a study using the OWASP WSTG that can provide a comprehensive picture of the security conditions of campus information systems and offer an overall strategy for maintaining and optimizing the security of the systems being studied.

This study is conducted to examine and measure aspects of system security through a vulnerability assessment analysis that follows the OWASP Web Security Testing Guide framework, in order to construct an advanced strategy to maintain and optimize it. The research is conducted by performing a case study on eight web-based information systems in a higher education institution so that the security condition of each website, as well as the overall characteristics of the systems, can be discovered. This study is expected to contribute to the understanding of how security controls for web-based information systems can benefit adopting organizations, especially higher education institutions.

2. LITERATURE REVIEW

2.1. VULNERABILITY ASSESSMENT IN WEB APPLICATIONS

One form of security risk in web applications is the exploitation of existing vulnerabilities, and thus it is important to perform assessments on them (Daniswara et al., 2020; Subandi & Sugara, 2022). The execution of said assessments should be performed by following a certain guide or framework in order to produce data that is unbiased, objective, and useful (Korczyński & Noroozian, 2023). It also needs to be performed frequently so that new vulnerabilities that may appear in the future can be taken care of as soon as possible before any damage has occurred (Higuera et al., 2020).

Note that vulnerability assessment differs from penetration testing. Vulnerability assessment is horizontal security testing, i.e., the tests are performed widely across various components in the system. In contrast, penetration testing is more vertical, where the tests are performed in-depth on one specific system part or component. In other words, a vulnerability assessment can show us how many security gaps and vulnerabilities are contained in the system, while penetration testing will show how bad the effects of a security gap being exploited would be (Ravindran & Potukuchi, 2022).

2.2. OWASP WEB SECURITY TESTING GUIDE

One of the available frameworks for security audits on web applications is the one created by the Open Worldwide Application Security Project (OWASP), namely the Web Security Testing Guide (WSTG) (Rafeli et al., 2022). OWASP WSTG is often preferred over other frameworks because it is free, open-source, actively maintained, supports automation, and can also be used as an instrument for manual penetration tests (Darojat et al., 2022; Kusuma, 2022).

The OWASP WSTG v4.2 documentation describes a series of web security controls in 12 discussion categories (see Table 1). By following the existing guides, security audits can be carried out in a structured manner, making it clear what tests need to be done, what the objectives are for each test, and what the test procedures are.

Table 1. Testing Categories of OWASP WSTG v4.2

Reference Number	Testing Categories
4.1	Information Gathering
4.2	Configuration and Deployment Management Testing
4.3	Identity Management Testing
4.4	Authentication Testing
4.5	Authorization Testing
4.6	Session Management Testing
4.7	Input Validation Testing
4.8	Testing for Error Handling
4.9	Testing for Weak Cryptography
4.10	Business Logic Testing
4.11	Client-side Testing
4.12	API Testing

2.3. COMMON VULNERABILITY SCORING SYSTEM

The Common Vulnerability Scoring System (CVSS) is a framework that provides a methodology to assess vulnerabilities (FIRST, 2019) so that the assessment can be performed through a measurement mechanism that is standard and consistent (Jiang & Atif, 2022).

CVSS divides its vulnerability assessment scheme into three groups: the Base Metric, Temporal Metric, and Environmental Metric. Each metric is analyzed separately, and commonly, only the Base Metric is published (see Table 2). This is because the Base Metric assesses the severity level of a vulnerability based on its intrinsic characteristics, which are measured in a constant way by assuming the worst possible impact. Thus, the final score it produces does not change over time and is valid in all environments. This differs from the Temporal and Environmental Metrics, whose final scores can change over time depending on several factors, such as the existence of exploitation techniques, the availability of solutions, and the importance of the impacted assets. Based on these definitions, the Base Metric is mandatory to be assessed in CVSS, while the Temporal and Environmental Metrics are optional but highly recommended to produce a more precise metric, especially for audits conducted through white-box or gray-box testing.

Table 2. Rating Scale of Vulnerability's Severity in CVSS

CVSS Base Score	Rating
0	None
0.1 – 3.9	Low
4.0 – 6.9	Medium
7.0 – 8.9	High
9.0 – 10.0	Critical

3. RELATED RESEARCH

Below are some previous research related to carrying out security audits to web-based security systems, as shown in Table 3.

Table 3. Related Research

Research Title	Author	Research Results	How It Differs From This Research
Penetration Testing to Detect The Vulnerability of Campu Information System	Sahren et al., 2019	Providing list of vulnerabilities found in the system through automatic scanning using Acunetix application, as well as its severity level.	This research follows the OWASP WSTG framework, and performing manual inspections besides from automatic scanning.
Testing for Information Gathering Using OWASP Testing Guide v4 (Case Study : Udayana University SIMAK-NG Application)	Daniswara et al., 2020	Providing the result list of Information Gathering by following the OWASP Testing Guide v4 in the description of "0", "1", or "-1".	This research follows the newer guide, which is the OWASP Web Security Testing Guide v4.2.

Research Title	Author	Research Results	How It Differs From This Research
Testing on Website Security Gaps Using Penetration Testing Techniques and the OWASP Top 10 Method on SIM XXX Website	Rosaliah et al., 2021	Providing list of OWASP Top 10 vulnerabilities that was found in the system, along with its CVSS scores and percentage of occurances.	This research audits all of the security aspects based on OWASP WSTG, not only the OWASP Top 10.
Security Gap Testing Using the OWASP Web Security Guide (WSTG) Method on XYZ Website	Rafeli et al., 2022	Providing result list of OWASP WSTG testings with the description of "Found" or "Not Found".	This research implements web security metric on the vulnerabilities that has been found.

4. RESEARCH METHOD

The security analysis conducted in this study is a vulnerability assessment performed on the web objects through black-box testing and is limited to user accounts with student or member roles. The method used in the study is a mix of qualitative and quantitative approaches with a case study design.

The qualitative method is a type of research where findings are obtained through the researcher's understanding or interpretation of what they encounter in the research object, without statistical procedures (Fiantika et al., 2022). The qualitative aspect of the vulnerability assessment in this study involves implementing the OWASP WSTG v4.2 framework. A series of security tests were conducted on the web objects, and a final conclusion was drawn based on the findings to determine if each test passed or failed. While the OWASP WSTG v4.2 framework explains web system security controls across 12 categories, this research focused on only four: Information Gathering, Configuration and Deployment Management Testing, Session Management Testing, and Client-side Testing.

Furthermore, each testing point that was declared "failed" (i.e., where vulnerabilities were found) was measured quantitatively through the Common Vulnerability Scoring System (CVSS) v3.1. This involved analyzing its exploitability, scope, and impact vectors to obtain a final CVSS Base Score, which indicates the severity level of the vulnerability. The quantitative method itself is a procedure to collect, analyze, and interpret data to gain measurable information that can be used for drawing conclusions and making decisions (Santoso & Madiistriyatno, 2021).

The population of this research is all web-based information systems at the higher education institution XYZ under the domain xyz.ac.id. Through a case study approach, eight information systems were chosen as samples. Four of these represent systems supporting academic activities with non-CMS framework-based infrastructures, while the other four represent non-academic information systems with CMS-based infrastructures. The case study approach involves an in-depth study of one case to generalize to a larger set of cases. This can be implemented observationally or experimentally, qualitatively or quantitatively, and with a small or large sample size (Rustendi, 2022).

5. RESULT AND DISCUSSION

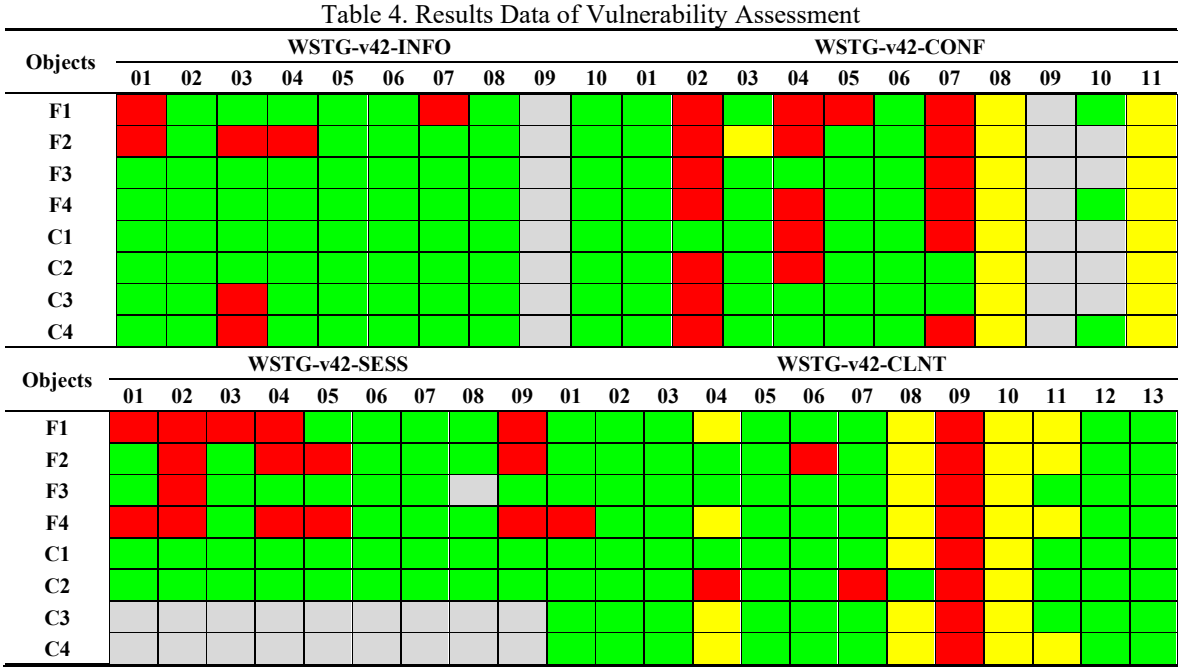
5.1. RESULT

Each test point of OWASP WSTG v4.2 conducted in this study was documented in the OWASP WSTG Testing Checklist, where all progress and findings were recorded, as well as the final conclusion of whether the test passed or failed. There are four types of final conclusions:

a. Not Started: The default status where a test has not been started, or the test is impossible to perform for some reason.
b. Pass: No security holes or gaps were found in the tested components.
c. Issues: Security holes or gaps were found during the tests.
d. Not Available: Tests have been performed and no security gaps were found, but a final conclusion of Pass or Issues cannot yet be drawn because further checks are necessary due to the limitations of black-box testing.

Table 4 below shows the overall data from the vulnerability assessment. For visual clarity in the table, the "Not Started" status will be marked in gray, "Pass" in green, "Issues" in red, and "Not Available" in yellow. Web objects in the campus academic support information system group (based on a non-CMS framework) are marked with the initial "Fn". Web objects in the non-academic information system group (based on a CMS) are marked with "Cn", where "n" is a number.

Table 4. Results Data of Vulnerability Assessment



Several tests have the status of "Not Started." For example, WSTG-v42-CONF-09 ("Test File Permissions") could not be run because it requires manual inspection of file and folder permissions on the server (e.g., via ls or namei commands on Linux), which was not possible through the black-box testing method conducted in this study. Most other tests were successfully implemented and had varying results of "Not Available," "Pass," or "Issues."

Furthermore, tests with the status of "Issues" will be analyzed further to determine their severity. The other statuses will not be discussed in more depth because no vulnerabilities were found in them. This analysis was carried out by calculating the Base Metric value from CVSS v3.1 by compiling vectors of exploitability, scope, and impact contained in each vulnerability. According to its calculation formula, if a vulnerability has no impact on the system, it will have a value of zero. Meanwhile, a vulnerability that has an impact will have a value greater than zero, with the specific value depending on its exploitability vector and scope.

Table 5. Base Metric Values of Found Vulnerabilities (CVSS 3.1)

Objects	WSTG-v42-INFO											WSTG-v42-CONF									
	01	02	03	04	05	06	07	08	09	10	01	02	03	04	05	06	07	08	09	10	11
F1	0						8.6					8.6		9.3	5.3		3.4				
F2	0		0	0								0		0			3.4				
F3												5.8					3.4				
F4												5.8		5.4			3.4				
C1														0			3.1				
C2												0		0							
C3			0									0									
C4			0									0					3.4				

Objects	WSTG-v42-SESS									WSTG-v42-CLNT											
	01	02	03	04	05	06	07	08	09	01	02	03	04	05	06	07	08	09	10	11	12	13
F1	0	6.1	6.9	6.1					6.1									3.4				
F2		6.1		6.1	6.5				6.1						4.7			3.4				
F3		6.1																3.4				
F4	0	6.1		6.1	6.5				6.1	6.1								3.4				
C1																		3.4				
C2													4.7			0		3.4				
C3																		3.4				
C4																		3.4				

As shown in Table 5, the vulnerabilities found in this research have varying severity weights. All five CVSS v3.1 severity types were found: None, Low, Medium, High, and Critical, with each occurring at a different rate.

5.2. DISCUSSION

Within the four OWASP WSTG v4.2 test categories included in this study's scope, there are a total of 43 sub-tests. Of those 43 sub-tests, two could not be carried out, 22 were declared "Pass" (no vulnerabilities found), and the remaining 19 had an "Issues" status for one or more web objects. The test results are visualized in Figure 1.

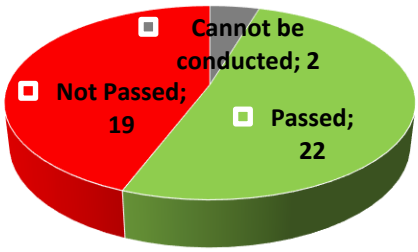


Figure 1. The Overall OWASP WSTG Test Results

For each web object tested, the one with the highest number of discovered vulnerabilities can be identified by counting the number of tests with a final status of "Issues." However, because the number of tests that were successfully carried out on each web object differs, it is more appropriate to compare the data as a ratio: the number of tests with an "Issues" result versus the total number of tests that could be run. An analysis of this ratio can be seen in Figure 2 below.

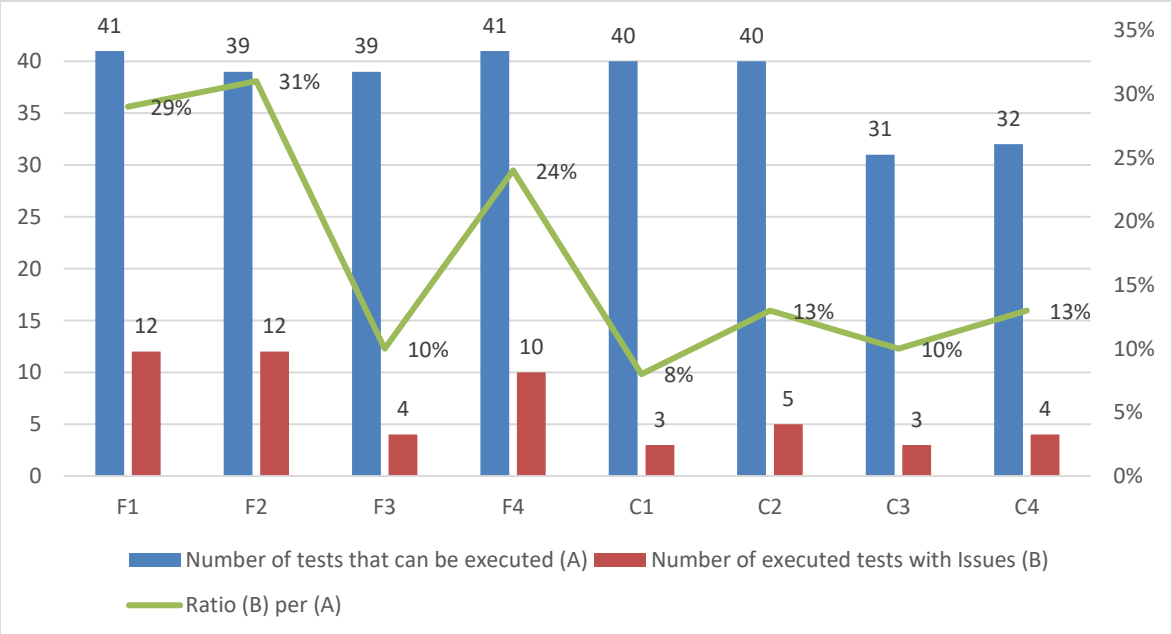


Figure 2. Vulnerability Discovery Ratio for Executed Tests

In terms of raw numbers, the F1 and F2 websites have the highest number of vulnerabilities, with 12 each. However, between the two, the ratio for F2 is slightly higher at 31%, compared to F1, which has 29%. Overall, the information systems supporting academic activities (based on non-CMS frameworks) were found to be more vulnerable, with a vulnerability ratio of 10%-31%, compared to the CMS-based non-academic

information systems, whose ratio was only 8%-13%.

Building a system using a framework gives developers relatively more freedom, but this can cause security problems if the development process is not accompanied by appropriate security controls. This is consistent with research by Ghozali et al. (2019), who found that a system built using a framework does not guarantee that it will be free from security holes.

The results of the CVSS Base Metric analysis of the discovered vulnerabilities produced varying final scores. A single vulnerability found on two different websites could have different CVSS scores; as in the WSTG-v42-CONF-02 test, where F1 had a value of 8.6 while F2 had a value of zero. These differences can occur due to variations in the weight of the exploitability, scope, and impact vectors contained in the vulnerabilities of each website. At its lowest level, a vulnerability can have a value of zero, meaning it is considered to have no severity because it does not have any impact on the confidentiality, integrity, or availability of the system. Figure 3 below presents the distribution of the web objects' CVSS scores at each severity level.

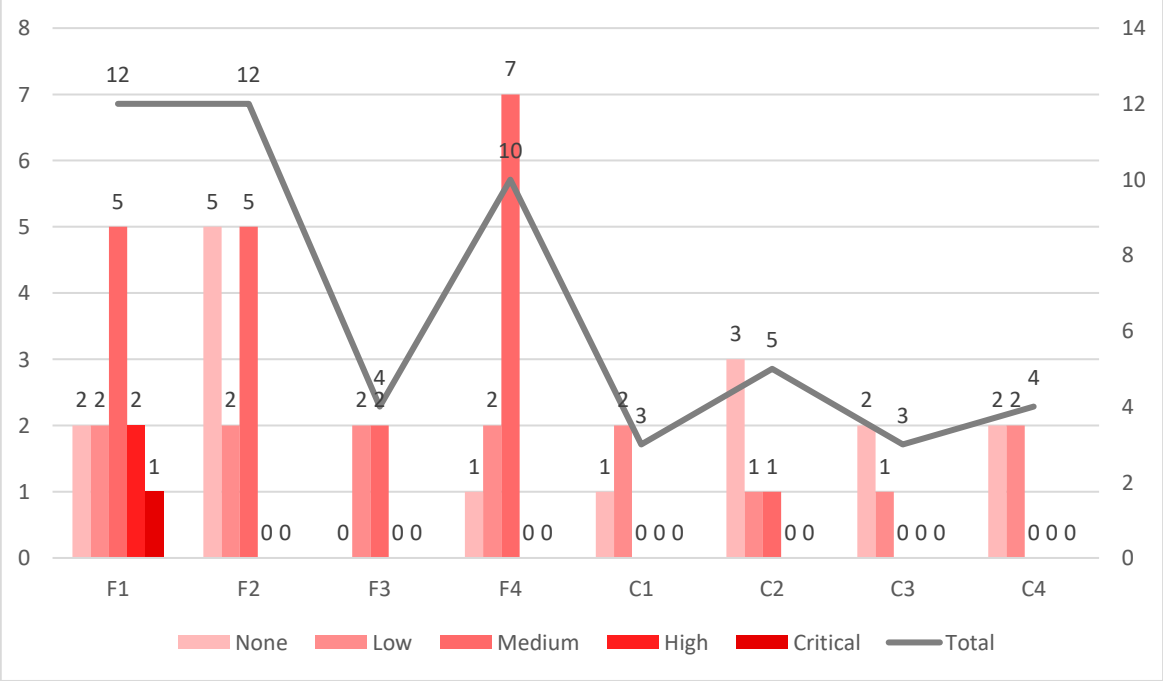


Figure 3. Distribution of Web Objects' CVSS Scores Based on The Severity Level

Looking at Figure 3, F1 is the only web object with a vulnerability that reaches the High and Critical severity levels. On the other hand, C1, C3, and C4 have maximum severity scores only at the Low level. Overall, it can be seen that academic support information systems based on non-CMS frameworks tend to have vulnerabilities with higher severity weights (Medium to Critical) compared to non-academic information systems based on CMS, which are primarily at the Low to Medium level.

Looking back at the previous discussion, these findings are consistent: the non-CMS framework-based academic support system group has a higher vulnerability discovery ratio and greater vulnerability severity. This means that, in general, it has more security vulnerabilities than the CMS-based non-academic systems.

Table 6. Order of Web Objects with The Highest CVSS Score

Web Objects	Highest CVSS Score	Web Objects	Highest CVSS Score
F1	9.3 (*Critical*)	C1	4.7 (*Medium*)
F2	6.5 (*Medium*)	C2	3.4 (*Low*)
F3	6.1 (*Medium*)	C3	3.4 (*Low*)
F4	6.1 (*Medium*)	C4	3.4 (*Low*)

From all the data above, the highest CVSS score for each web object can be identified. These web objects can then be sorted from the most to the least severe based on this score. Table 6 presents this sequence.

As seen in Table 6, the non-CMS framework academic group has maximum CVSS levels that are mostly Medium, except for F1, which reaches Critical. In contrast, the CMS-based non-academic group mostly has maximum scores at the Low level, with the highest reaching only the Medium level. From the eight web objects tested, it is clear that F1 is the website that requires the highest priority for follow-up actions. Other web objects are, of course, no less important to address, because if vulnerabilities in a system are ignored, they will always be susceptible to the threat of cyberattacks.

The security conditions of the information systems in the higher education institution obtained in this study can also be contextualized by comparing them with other studies. Mateus & Serrão (2021) conducted security testing on 10 university websites and found that five web objects had vulnerabilities at the Medium level and five others at High. There is also the study by Sahren et al. (2019), who tested three campus information systems with final results showing two systems had vulnerabilities at the Medium severity level and the other one at High. Compared to these two studies, the findings obtained in this study show relatively better results proportionally, where of the eight web objects tested, only F1 has a vulnerability that reaches the Critical severity level, while the other seven websites only have maximum severity levels of Low or Medium. It should be noted that the two comparative studies above have different security test designs compared to this study, so any deeper comparison should be interpreted contextually.

Based on all the tests that have been conducted, follow-up recommendations were prepared for the 19 vulnerable sub-tests (out of 43 total OWASP WSTG sub-tests). These recommendations highlight several important security principles:

a. System infrastructure needs to be designed while anticipating potential future cyberattacks. This can be done by installing a firewall, Intrusion Detection System (IDS), Intrusion Prevention System (IPS), Honeypot, and so on.
b. Each component of the entire system needs to be configured properly. This includes setting access permissions for server files and directories, setting user authorization for each URL, configuring cookie attributes, and implementing Content Security Policy (CSP) and HTTP Strict Transport Security (HSTS).
c. It is not uncommon for sensitive internal system information to leak, such as specific component versions, overly detailed error descriptions, and even debugging messages or code. For this reason, information posted online in the system must be reviewed for its sensitivity. This includes information on web pages displayed directly to users, as well as indirectly through response headers, HTML metadata, and JavaScript code or comments.
d. Always validate user inputs before processing them further. Without strict and proper validation, users can insert injections that can lead to Cross-Site Scripting (XSS) or Open Redirection. User input should also be ignored in processes that run entirely on the server side, such as session token generation, so that exploitation of vulnerabilities like Session Fixation or Client-side Resource Manipulation can be avoided.

6. CONCLUSION

Based on the discussion in this research, the group of CMS-based information systems was found to be less vulnerable than those based on non-CMS frameworks, both in terms of vulnerability discovery ratio and severity level. The follow-up recommendations indicate that it is not enough to apply system security controls only after a vulnerability is discovered. Instead, security controls should be implemented as early as possible in the system infrastructure design process and must be applied routinely and in a measurable manner throughout the entire lifecycle: from design, development, and testing to production and post-production.

This research contributes to the higher education institution that owns the tested systems and provides other researchers with an in-depth picture of how security controls can be implemented in web-based information systems. For further research, OWASP WSTG security testing could be conducted on categories other than the four examined in this study, or even on all 12 OWASP WSTG categories. The selection of tested systems could also be different, for example, by comparing systems with similar functions or frameworks across several higher education institutions simultaneously.

REFERENCES

Andria, A. (2020). Website security gap analysis using WEBPWN3R tools at Kali Linux. *Generation Journal, 4*(2), 69–76.

Anggraeni, D. P., Zen, B. P., & Pranata, M. (2022). Security analysis on websites using the information system assessment framework (ISSAF) and Open Web Application Security version 4 (OWASPv4) using the penetration testing method. *Jurnal Pertahanan: Media Informasi Ttg Kajian & Strategi Pertahanan Yang Mengedepankan Identity, Nasionalism & Integrity, 8*(3), 497–506.

Awlarijal, A. N., Almaarif, A., & Budiono, A. (2020). Vulnerability assessment for basic data of education website in regional government X–a black box testing approach. *FoITIC*, 163–168.

Aziz, M. A. (2021). Vulnerability assesment untuk mencari celah keamanan web aplikasi e-learning pada universitas XYZ. *Journal of Engineering, Computer Science and Information Technology (JECSIT), 1*(1).

Daniswara, R. R., Sasmita, G. M. A., & Pratama, I. (2020). The testing for information gathering using OWASP testing guide v4 (case study: Udayana University SIMAK-NG application). *JITTER: Jurnal Ilmiah Teknologi Dan Komputer, 1*(1).

Darojat, E. Z., Sediyono, E., & Sembiring, I. (2022). Vulnerability assessment website e-government dengan NIST SP 800-115 dan OWASP menggunakan web vulnerability scanner. *JSINBIS (Jurnal Sistem Informasi Bisnis), 12*(1), 36–44.

Fiantika, F., Wasil, M., Jumiyati, S. R. I., Honesti, L., Wahyuni, S. R. I., Mouw, E., Mashudi, I., Hasanah, N. U. R., Maharani, A., & Ambarwati, K. (2022). *Metodologi penelitian kualitatif*. PT. Pustaka Pelajar.

FIRST. (2019). *Common vulnerability scoring system version 3.1 specification document revision 1*. https://www.first.org/cvss/

Ghozali, B., Kusrini, K., & Sudarmawan, S. (2019). Mendeteksi kerentanan keamanan aplikasi website menggunakan metode owasp (open web application security project) untuk penilaian risk rating. *Creative Information Technology Journal, 4*(4), 264–275.

Guntoro, G., Costaner, L., & Musfawati, M. (2020). Analisis keamanan web server open journal system (OJS) menggunakan metode issaf dan owasp (studi kasus ojs universitas lancang kuning). *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika), 5*(1), 45–55.

Higuera, J. R. B., Higuera, J. B., Montalvo, J. A. S., Villalba, J. C., & Pérez, J. J. N. (2020). Benchmarking approach to compare web applications static analysis tools detecting OWASP top ten security vulnerabilities. *Computers, Materials and Continua, 64*(3).

Jiang, Y., & Atif, Y. (2022). Towards automatic discovery and assessment of vulnerability severity in cyber–physical systems. *Array, 15*, 100209.

Korczyński, M., & Noroozian, A. (2023). *Security reputation metrics*. ArXiv Preprint ArXiv:2302.07172.

Kusuma, G. (2022). Implementasi OWASP ZAP untuk pengujian keamanan sistem informasi akademik. *Jurnal Teknologi Informasi: Jurnal Keilmuan Dan Aplikasi Bidang Teknik Informatika, 16*(2), 178–186.

Listartha, I. M. E., Mitha, I. M. A. P., Arta, M. W. A., & Arimika, I. K. W. Y. (2022). Analisis kerentanan website SMA Negeri 2 Amlapura menggunakan metode OWASP (open web application security project). *Jurnal Sistem Informasi Dan Sistem Komputer, 7*(1), 23–27.

Maharani, M. Z., Andrian, H. R., & Ismail, S. J. I. (2017). Analisis keamanan website menggunakan metode scanning dan perhitungan security metriks. *EProceedings of Applied Science, 3*(3).

Mateus, E., & Serrão, C. (2021). *Vulnerability assessment of Angolan university web applications*.

Mu'min, M. A., Fadlil, A., & Riadi, I. (2022). Analisis keamanan sistem informasi akademik menggunakan open web application security project framework. *Jurnal Media Informatika Budidarma, 6*(3), 1468. https://doi.org/10.30865/mib.v6i3.4099

OWASP. (2020). *Web security testing guide v4.2*. OWASP.Org. https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/00-Introduction_and_Objectives/README

OWASP Foundation. (2021). *OWASP web security testing guide (WSTG) v4.2*. https://owasp.org/www-project-web-security-testing-guide/

Putra, F. G., & Soewito, B. (2022). Measurement of security system performance on websites of personnel information systems in government using common vulnerability scoring system. *Jurnal Pendidikan Tambusai, 6*(1), 2949–2957.

Rafeli, A. I., Seta, H. B., & Widi, I. W. (2022). Pengujian celah keamanan menggunakan metode OWASP web security testing guide (WSTG) pada website XYZ. *Informatik: Jurnal Ilmu Komputer, 18*(2), 97–103.

Ravindran, U., & Potukuchi, R. V. (2022). A review on web application vulnerability assessment and penetration testing. *Review of Computer Engineering Studies, 9*(1), 1–22. https://doi.org/10.18280/rces.090101

Rosaliah, Y. T. A., Jayanta, J., & Hananto, B. (2021). Pengujian celah keamanan website menggunakan teknik penetration testing dan metode OWASP TOP 10 pada website SIM xxx. *Prosiding Seminar Nasional Mahasiswa Bidang Ilmu Komputer Dan Aplikasinya, 2*(2), 752–761.

Rustendi, T. (2022). Pendekatan kuantitatif dalam studi kasus pada penelitian bidang akuntansi. *Jurnal Akuntansi, 17*(1), 24–37.

Sahren, S., Dalimuthe, R. A., & Amin, M. (2019). Penetration testing untuk deteksi vulnerability sistem informasi kampus. *Prosiding Seminar Nasional Riset Information Science (SENARIS), 1*, 994–1001.

Santoso, I., & Madiistriyatno, H. (2021). *Metodologi penelitian kuantitatif*. Indigo Media.

Subana, B., Fadlil, A., & Sunardi, S. (2020). Web server security analysis using the OWASP mantra method. *Jurnal Mantik, 4*(1), 107–116.

Subandi, K., & Sugara, V. I. (2022). Analisa serangan vulnerabilities terhadap server selama periode WFH di masa pandemi Covid-19 sebagai prosedur mitigasi. *Jurnal Asiimetrik: Jurnal Ilmiah Rekayasa & Inovasi*, 125–132. https://doi.org/10.35814/asiimetrik.v4i1.3127

Tania, A. M., Setiyadi, D., & Khasanah, F. N. (2018). Keamanan website menggunakan vulnerability assessment. *Informatics for Educators and Professional: Journal of Informatics, 2*(2), 171–180.

Yudiana, Y., Elanda, A., & Buana, R. L. (2021). Analisis kualitas keamanan sistem informasi e-office berbasis website pada STMIK Rosma dengan menggunakan OWASP top 10. *CESS (Journal of Computer Engineering, System and Science), 6*(2), 185–191.