Journal of Intelligent Decision Support System (IDSS) 8 (1) (2025), pp. 1-8

Published by: Institute of Computer Science (IOCS)

Journal of Intelligent Decision Support System (IDSS)

Journal homepage: www.idss.iocspublisher.org

Website security analysis using penetration testing method

Siti Anisah[1], Suwaebatul Aslamiyah[2]

Department of Informatics Engineering, Universitas Indraprasta PGRI, Indonesia

Article Info

Article history:

Received Dec 27, 2024
Revised Jan 05, 2025
Accepted Jan 14, 2025

Keywords:

Network security;
Vulnerability;
Penetration;
Testing;
Website.

ABSTRACT

Website security is one of the main focuses in information system management, especially with the increasing cyber threats that can damage the integrity and confidentiality of data. One way to identify security gaps through penetration testing is widely used using automated tools to improve efficiency and accuracy. Identifying potential vulnerabilities such as SQL injection, Cross-Site Scripting (XSS), and configuration failures in This study involved implementing automated tools on several website tests, where the test results were then analyzed to determine potential security risks. The study found vulnerabilities in the form of Application Error Disclosure, Content Security Policy (CSP), hidden files found, servers leaking information via x-power-by, servers leaking version information via the server, x-content-type-options headers missing, and user agent fuzzier These findings contribute to efforts to improve the quality of automated security testing, as well as optimizing potential threat mitigation actions. Evaluate and disable components that are not needed in production, Disable or restrict closing the "X-Powered-By" and "Server" headers, Check for different responses based on User Agent, and use the HTTPS protocol throughout the application to improve its security.

Corresponding Author:

Siti Anisah
Teknik Informatika,
Universitas Indraprasta PGRI,
Jl.Nangka Raya No.58, Tanjung Barat, Jagakarsa, South Jakarta, DKI Jakarta, 12530, Indonesia
Email: Anis.siti.ssa@gmail.com

Introduction

The development of internet technology and the use of web-based applications have changed the way many companies operate and provide their services to customers. An application is a media information service or collection of pages that displays data information (Rizkayanti & W, 2023). The increasing use of the internet in addition to providing convenience for its users is also accompanied by negative impacts that arise in the form of security threats. Based on Internetworldstate data, there are 212.35 million or 76.3% of the total population of Indonesia who actively use the internet (Burhani & Priyawati, 2024). A website or site is a collection of pages used to display text information, still or moving images, animations, sounds, and/or a combination of all of them (Antasari, 2020). Web applications are now the main platform for various services, including academic information systems, e-commerce, online banking, and social media applications. As a result, the amount of sensitive data handled by web applications is increasing. Web application security is also a very important aspect, because threats to this sensitive data can cause significant financial losses, reputational damage, and

even legal sanctions. Therefore, database security becomes very important to protect the sensitive information stored in it (Ujung et al., 2023).

Threats to web applications are increasingly diverse and complex (Sandag et al., 2020), with attacks such as SQL injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), and broken authentication continuing to grow and become more sophisticated. Information security involves protecting data from unauthorized access, modification, or destruction (Natanael et al., 2024). Therefore, it is important for developers and organizations to proactively evaluate and test the security of their web applications. With the existence of security gap testing, it can be a solution that is used as a benchmark to improve the website security system in the future (Rosaliah, Jayanta, et al., 2021).

A tester must have knowledge about testing a network or system (Guntoro et al., 2020). Penetration testing is one of the effective methods to identify and exploit vulnerabilities in web applications. Penetration testing uses the methods and means used by cybercriminals to find these vulnerabilities, but is authorized to do so. This means that unlike cybercriminals, penetration testers have the approval/permission of the organization being tested (Fachri et al., 2021). The main purpose of penetration testing is to identify vulnerabilities that can be exploited by attackers, organizations can take steps to fix and secure their systems before the vulnerabilities can be exploited by actual attackers (Fadhli, 2024). Penetration testing simulates attacks that might be carried out by attackers to expose weaknesses in the security system (Hardani & Ramli, 2022). One of the tools widely used in penetration testing is OWASP ZAP (Zed Attack Proxy), which is an open-source testing tool designed to find vulnerabilities in web applications automatically and manually. ZAP supports various types of testing, including SQL injection scanning, XSS, CSRF, and authentication analysis. Vulnerabilities in web applications usually occur at the database level or the network level (Herman et al., 2023).

This study aims to test the security of web applications using OWASP ZAP on an academic information system. Academic information system is a system used by educational institutions that is utilized to improve services (Prihandoyo, 2020). Academic information system has a lot of important user information that can be the reason academic information system becomes the target of attack. This academic information system is tested to explore various types of vulnerabilities and to assess the extent to which tools such as ZAP can help in identifying security issues and providing recommendations for improvement based on test results to strengthen web application security.

Method

This study uses OWASP ZAP (Zed Attack Proxy) as a tool to conduct web application security testing. ZAP is an application for scanning to find vulnerabilities in a web application easily (Christina Sari et al., 2024), specifically the type of cybersecurity (Sulisnawati, 2023). The research process is divided into several stages, which involve preparation, application scanning, analysis of results, and formulation of recommendations. The steps taken in this study are as follows:

Figure 1. research stages

This study conducted security testing on academic information systems to identify their various vulnerabilities. Information security strategythrough rapid and accurate vulnerability identification, proactive elimination of identified risks, implementation of corrective actions, and enhancement of IT knowledge (Umasugi et al., 2024). ZAP provides various automatic and manual features that allow for comprehensive and efficient vulnerability detection (Setyaningrum, 2023). Vulnerability analysis using the OWASP ZAP tool will produce several weakness factors that will be tested (Simanjuntak et al., 2024). This testing process includes spidering, passive scanning, active scanning, and manual testing, which allows researchers to identify vulnerabilities such as SQL injection, XSS, CSRF, and authentication-related issues. SQL Injection is a security attack technique on web applications that allows attackers to inject malicious SQL code into the input expected by the

application(Anugrah Utama & Supardi, 2024). The test results provide very useful insights to improve application security by providing recommendations for mitigating its vulnerabilities. With the Owasp-Zap tool, vulnerability detection on the lab.scarpe website can be carried out and analysis of the vulnerability results can also be concluded so that it can be used as a reference for improving the existing lab scarpe system (Hasibuan et al., 2023).

To ensure the validity of the results in software testing, it's crucial to set up the test environment in a controlled and systematic manner. Here's a comprehensive answer on how to set up the test environment: (a) The hardware used for testing should be as close as possible to the production environment in terms of specifications and performance. This includes servers, workstations, network devices, and any peripherals that might be part of the end-user setup. (b) The test environment should replicate the software versions, operating systems, and configurations found in the production environment. This includes databases, third-party applications, and middleware, ensuring that the test system mirrors the actual system as closely as possible. (c) The network configuration should reflect the actual deployment scenario. This includes setting up firewalls, VPNs, proxies, and ensuring that network latency, bandwidth, and security configurations are similar to those in a live environment. (d) The test environment should use data that mirrors the kind of data expected in production. (e) Ensure that the environment remains consistent throughout the testing phase. Any changes in the configuration or hardware/software components can impact the results. (f) The tools used for testing (e.g., testing frameworks, automation tools, load testing tools, and monitoring software) must be compatible with the environment and accurately reflect the desired test cases. (g) Continuous monitoring of the test environment helps ensure that there are no unexpected changes or errors during testing that could affect the results. Monitoring tools can check for server performance, response times, network speed, and other relevant metrics that could impact the reliability of the test results. (h) The environment should be configured to allow tests to be repeated with the same conditions. This ensures that if tests need to be run multiple times (for regression, comparison, etc.), the results will be comparable, and any changes can be tracked effectively. (i) Any variables outside of the test focus (such as time of day, server load, or background applications) should be controlled or accounted for in the environment setup to minimize their impact on the validity of the results.

Results and Discussions

Web Application Selection and Testing Environment

The application used in this study is an academic information system application that allows users to make educational payment transactions and store personal information. Researchers have collected vulnerability findings and conducted attack simulations against the vulnerabilities found (Purnomo & Chusyairi, 2024). OWASPZap has an intuitive graphical user interface (GUI), making it easy to use by developers and security researchers (Nurelasari et al., 2024). This application was chosen because of the large amount of sensitive data processed, making it a potential target for cyber attacks. Testing was conducted in a testing environment that is similar to the application in real situations, but without involving actual user data to maintain privacy and testing ethics.

Figure 2. Sample website to be tested

Testing Tool Setup (OWASP ZAP)

Before testing begins, the first step is to configure and set up OWASP ZAP. ZAP can be used in two modes, namely automated and manual. In this study, a combination of both modes was used to conduct comprehensive testing. The settings made include: (a) Proxy Setup: ZAP is configured as a proxy between the web application and the browser to allow monitoring and analysis of HTTP/HTTPS traffic that occurs between the client and the server. (b) Spidering (Crawling): ZAP crawls the application to identify all existing pages and endpoints, and understand the structure of the application. This allows ZAP to gather information about potential attack points. (c) Auto-scan: ZAP auto-scan settings to automatically identify vulnerabilities based on common attack patterns such as SQL injection, XSS, CSRF, and others.

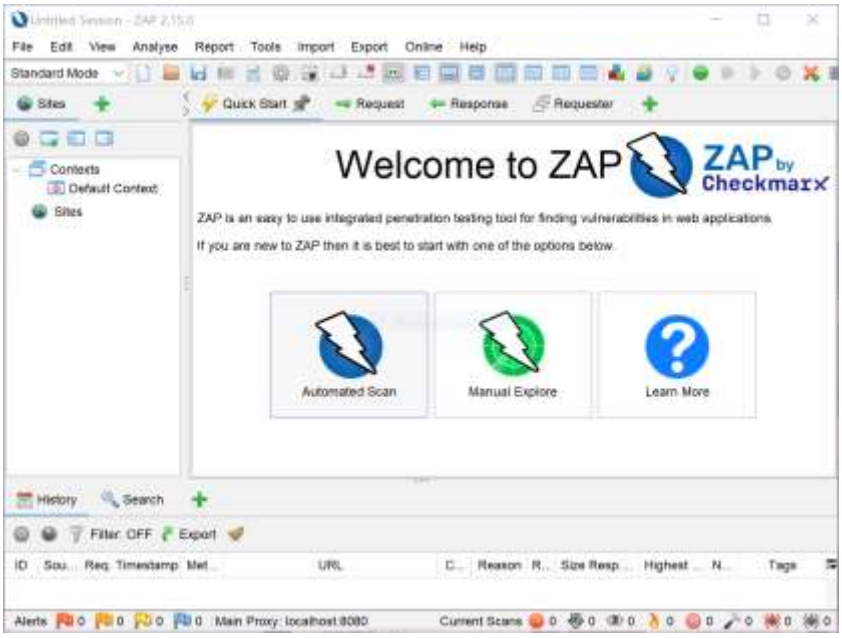

Figure 3. Website

The test result is an alert in the form of a flag with different colors depending on the level of vulnerability of the website. Details of the warning flag colors can be seen in the following table:

Table 1. Risk Level

Color	Risk Level
	High
	Medium
	Low
	Informational

Security Testing Process

In security testing using OWASP ZAP performs active scanning to analyze potential vulnerabilities such as SQLi, XSS, CSRF, and others. This tool checks input parameters, HTTP headers, and application configurations for potential exploitable gaps.

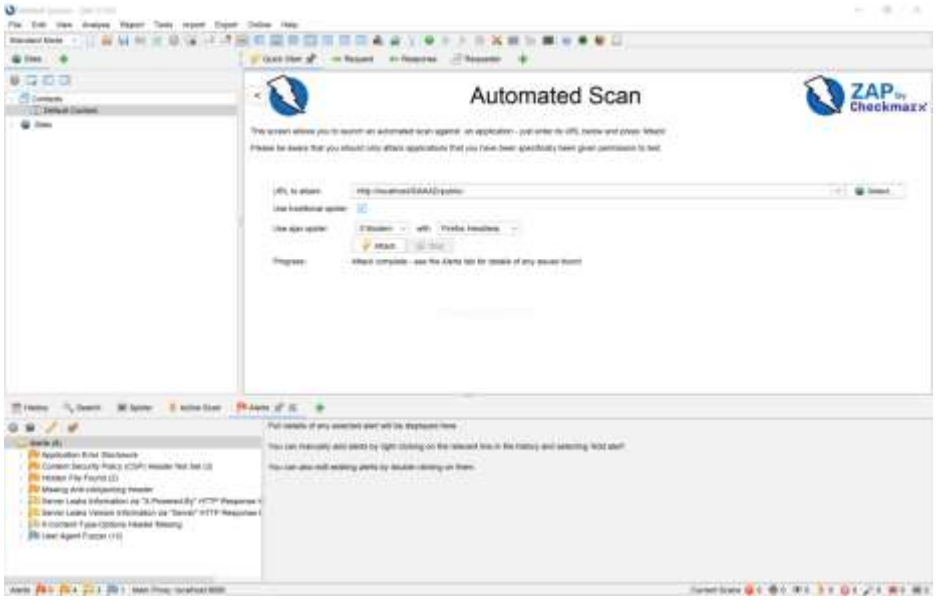

Figure 3. *Automated scan result*

From the results of the automated scan, system vulnerabilities were found with low and medium risk levels. These vulnerabilities include Application Error Disclosure, Content Security Policy (CSP), Hidden file found, server leaks information via x-powered-by, server leaks version information via server, and x-content-type-options header missing.

Analysis and Evaluation of Results

After performing a scan with OWASP ZAP, the test results are analyzed to identify and evaluate the vulnerabilities found. ZAP provides a detailed report that includes the type of vulnerability, location (affected page or endpoint), severity (e.g., high, medium, or low), and recommendations for remediation.

The findings from the ZAP report are then further evaluated to determine whether the attack could be exploited in the context of the application being tested. Researchers also classify the vulnerabilities based on their potential impact on application security, user data, and transaction integrity.

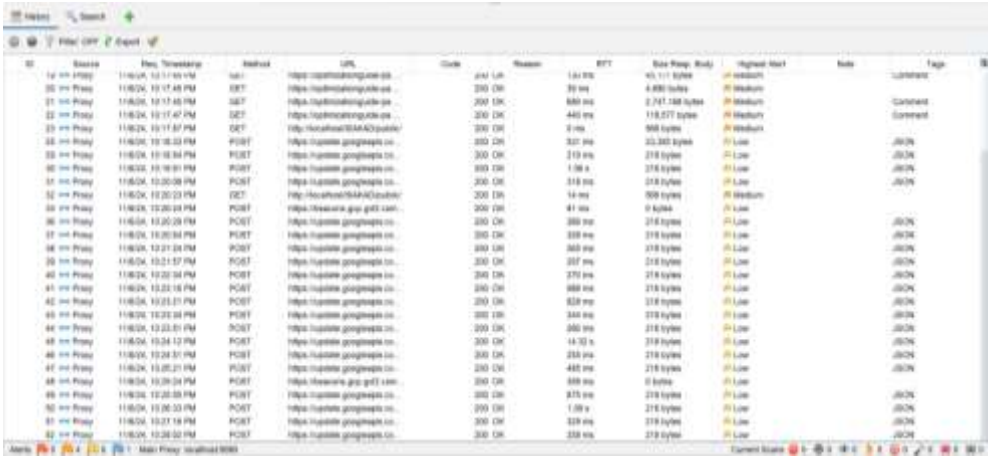

Figure 4. *vulnerability findings snippet*

Formulation of Security Recommendations

Based on the results of the analysis, mitigation steps and recommendations to fix the vulnerabilities are prepared. The recommendations focus on several things as shown in the following table:

Table 2. Formulation of Security Recommendation

No	Risk Level	Vulnerability	Solution
1	Medium	Application Error Disclosure	Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and not exposing them to the user.
2	Medium	Content Security Policy (CSP)	It's recommended to implement Content Security Policy (CSP) into your web application. Configuring Content Security Policy involves adding the Content-Security-Policy HTTP header to a web page and giving it values to control resources the user agent is allowed to load for that page.
3	Medium	Hidden file found	Consider whether or not the component is actually required in production, if it isn't then disable it. If it is then ensure access to it requires appropriate authentication and authorization, or limit exposure to internal systems or specific source IPs, etc.
4	Low	server leaks information via x-powered-by	Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.
5	Low	server leaks version information via server	Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details.
6	Low	x-content-type-options header missing	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.
7	Informational	user agent fuzzer	Check for differences in response based on fuzzed User Agent (eg. mobile sites, access as a Search Engine Crawler)

Conclusions

Based on the results of security testing conducted using penetration testing methods with automated tools, this study shows that the use of automated tools, such as OWASP ZAP, can effectively identify various vulnerabilities that are often found in web applications, such as SQL injection, Cross-Site Scripting (XSS), and server configuration issues that can be exploited by irresponsible parties. Although automated tools provide efficiency in detecting vulnerabilities, the findings from this test also show that manual analysis is still needed to handle more complex and specific problems. Therefore, website security testing does not only rely on automated tools, but also requires supervision and handling from experienced security professionals. Suggestions for future website improvements are Source Code Improvement and Error Management, implementation of Content Security Policy (CSP), Unnecessary Component Checks, Disabling Headers that Reveal Sensitive Information and Testing Based on User Agent. By implementing the recommended improvements, the website can avoid various security threats that have the potential to damage reputation and data integrity. This study provides important insights into the importance of regular security testing using automated tools and the implementation of better security practices in the development and management of web applications which is not only limited to testing academic information systems, but can also be used to test other websites. The findings of this study can significantly contribute to the development of IT security policies in the education sector in the following ways such as enhancing application and system security, implementing security testing, user education and training, including training on how to recognize cyber threats and how to use applications safely and implementing security standards by providing a basis for creating or updating it security standards in the education sector, such as student personal data management policies, protection against malware attacks, and protection of academic

data. The integration of these research findings can strengthen access control and authentication for critical systems in the education environment.

References

Antasari, I. W. (2020). Membuat Website/Blog Profesional sebagai Sarana Penyebaran Informasi Sekolah. *Journal Perpusnas*, *24 no.2*, 25–30. Https://Ejournal.Perpusnas.Go.Id/Mp/Article/View/10%0

Anugrah Utama, D., & Supardi, R. (2024). Analisis Keamanan Website Menggunakan PTES (Penetration Testing Execution And Standart). *Jurnal Media Infotama*, *20*(0736), 106–112. http://info.cern.ch.

Burhani, L. F., & Priyawati, D. (2024). ANALISIS PENGUJIAN KEAMANAN WEBSITE PENGELOLAAN INTERNET DESA KRAGAN MENGGUNAKAN METODE PENETRATION TESTING EXECUTION STANDARD (PTES). *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *9*(1), 307–319. https://doi.org/10.29100/jipi.v9i1.4455

Christina Sari, N., Solichan, A., Ansor, B., Putra Ramdani, A., Zainudin Al Amin, M., Khaira, M., & Rohman Riquelme Al Ubaidah, A. (2024). Deteksi Kerentanan SQL Injection pada Website Menggunakan Vulnerability Assessment. *Journal of Data Insights*, *2*(1), 9–17. https://doi.org/10.26714/jodi

Fachri, F., Fadlil, A., Riadi, I., Dahlan, A., Jln Soepomo, Y., & Artikel, I. (2021). Analisis Keamanan Webserver Menggunakan Penetration Test. *JURNAL INFORMATIKA*, *8*(2). http://ejournal.bsi.ac.id/ejurnal/index.php/ji

Fadhli, M. (2024). *Comprehensive Analysis of Penetration Testing Frameworks and Tools: Trends, Challenges, and Opportunities*. *4*(June), 15–22.

Guntoro, Costaner, L., & Musfawati. (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)*, *05*, 45–55.

Hardani, M. S., & Ramli, K. (2022). Perancangan Manajemen Risiko Keamanan Sistem Informasi Manajemen Sumber Daya dan Perangkat Pos dan Informatika (SIMS) Menggunakan Metode NIST 800-30. *JURIKOM (Jurnal Riset Komputer)*, *9*(3), 591. https://doi.org/10.30865/jurikom.v9i3.4181

Hasibuan, A. F., Tommy, & Handoko, D. (2023). Analisis Keretanan Website Dengan Aplikasi Owasp Zap. *Jurnal Ilmu Komputer Dan Sistem Informasi (JIRSI)*, *2*, 257. http://creativecommons.org/licenses/by-sa/4.0/

Herman, H., Riadi, I., Kurniawan, Y., & Rafiq, I. A. (2023). Analisis Keamanan Website Menggunakan Information System Security Asessment Framework(ISSAF). *Jurnal Teknologi Informatika Dan Komputer*, *9*(1), 126–136. https://doi.org/10.37012/jtik.v9i1.1439

Natanael, Y., Felicia, R., & Sakti, E. M. S. (2024). Analisis Keamanan Informasi Bagi Pengguna Website Menggunakan Kalilinux Melalui Teknik SQL Injection. *Jurnal Ilmiah Teknik Informatika …*, *25*(1), 123–132.

Nurelasari, E., Gumilang, D., & Farabi, A. (2024). ANALISIS KEAMANAN SISTEM WEBSITE MENGGUNAKAN METODE OPEN WEB APPLICATION SECURITY PROJECT (OWASP) PADA SIMANTEP.ID. *Jurnal Mahasiswa Teknik Informatika*, *8*(3), 3049–3054.

Prihandoyo, M. T. (2020). Unified Modeling Language (UML) Model Untuk Pengembangan Sistem Informasi Akademik Berbasis Web. *Jurnal Pengembangan IT (JPIT)*, *03, No.1*, 126–129.

Purnomo, M. D., & Chusyairi, A. (2024). Pengujian Keamanan Sistem Mengunakan Metode Penetration Testing di Website Diskominfostandi Kota Bekasi. *Jurnal Ilmiah Sistem Informasi*, *1*(1), 92–101. https://doi.org/10.69533

Rizkayanti, T., & W, Y. (2023). ANALISIS KEAMANAN WEBSITE SISTEM INFORMASI ADMINISTRASI KEPENDUDUKAN MENGGUNAKAN METODE VULNERABILITY ASSESMENT. *Teknologi Informatika Dan Komputer*, *1*(1), 1–9. https://doi.org/10.xxxxx

Sandag, G. A., Leopold, J., & Ong, V. F. (2020). Klasifikasi Malicious Websites Menggunakan Algoritma K-NN Berdasarkan Application Layers dan Network Characteristics Malicious Websites Classification Using K-NN Algorithm Based on Application Layers and Network Characteristics. *Cogito Smart Journal*, *4*(1).

Setyaningrum, I. (2023). PENGEMBANGAN APLIKASI MONITORING KEAMANAN UNTUK PENGUJIAN CELAH KEAMANAN APLIKASI LAPORAN PELAKSANAAN ANGGARAN BERBASIS WEBSITE DENGAN STANDARISASI OWASP. *Researchgate.Net*, 10115277.

Simanjuntak, C. P., Dyah Arsanti, U., & Sudarmana, L. (2024). ANALISIS KEAMANAN SISTEM MENGGUNAKAN METODE PENETRATION TESTING PADA WEBSITE. *SEMINAR NASIONAL AMIKOM*, 1236–1246. https://kekampus.umri.ac.id/.

Sulisnawati, N. (2023). Implementation of Open Web Application Security Project for Penetration Testing on Educational Institution Websites. *Jurnal Ilmiah Teknik Elektro Komputer Dan Informatika (JITEKI)*, *9*(2), 250–267. https://doi.org/10.26555/jiteki.v9i2.25987

Rosaliah, Y. T. A., Jayanta, & Hananto, B. (2021). Pengujian Celah Keamanan Website Menggunakan Teknik Penetration Testing dan Metode OWASP TOP 10 pada Website SIM xxx. In *Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA) Jakarta-Indonesia*.

Ujung, A. M., Irwan, M., & Nasution, P. (2023). Pentingnya Sistem Keamanan Database untuk melindungi data pribadi. *JISKA: Jurnal Sistem Informasi Dan Informatika*, *1*(2), 44. http://jurnal.unidha.ac.id/index.php/jteksis

Umasugi, M. R., Satra, R., Widya, A., & Gaffar, M. (2024). Analisis Keamanan Website dengan Metode Penetration Testing pada PT. PLN (Persero). *Literatur Informatika & Komputer*, *1*(3), 293–301. https://doi.org/10.33096/linier.vxix.xxxx