Journal of Deep Learning, Computer Vision and Digital Image Processing
E-ISSN: 2986-8939 P-ISSN: 2986-8920
Vol. 4, No. 3, September 2026
DOI. https://doi.org/10.61255/decoding.v4i3.1724

Security Vulnerability Analysis of an Employee Attendance Website Using Black-Box Penetration Testing and OWASP Top 10 2025

Rizqullah Raihan ✉
(Telkom University, Bandung, Indonesia)

Muhamad Irsan
(Telkom University, Bandung, Indonesia)

Yanuar Firdaus Arie Wibowo
(Telkom University, Bandung, Indonesia)

OPEN ACCESS

ARTICLE HISTORY
Received: June 23, 2026
Revised: July 21, 2026
Accepted: August 24, 2026

KEYWORDS
Employee attendance system;
OWASP Top 10 2025;
Penetration testing;
Vulnerability assessment;
Web security.

ABSTRACT

Purpose – This study evaluates the security posture of the PT. Jaka Kuasa Nusantara employee-attendance website, which supports account authentication, attendance recording, and face-capture functionality.

Methods – An authorized black-box vulnerability assessment and penetration-testing approach was conducted through information gathering, network and service reconnaissance, Nessus vulnerability scanning, and targeted manual verification. The observations were interpreted using the OWASP Top 10 2025 framework, the CIA Triad, and CVSS-oriented risk considerations.

Findings – Nessus identified HTTP configuration issues, cookies without HttpOnly and Secure attributes, an improper or missing Content-Type configuration, and an Apache 2.4.29 version requiring patch-status verification. Manual testing indicated that SQL injection was not successful in the tested login scenario, CAPTCHA and login-attempt limiting were present, and no SSRF weakness was observed in the tested scope.

Research implications – The assessment was limited to authorized, externally observable behavior and did not include destructive exploitation, source-code review, or full internal-infrastructure testing.

Originality – The study integrates automated vulnerability discovery with scenario-based manual validation and translates the results into a prioritized mitigation roadmap for an operational attendance application. The recommended actions include HTTPS and HSTS enforcement, security-header hardening, secure-cookie configuration, component patching, stronger authentication, centralized logging, and periodic retesting.

INTRODUCTION

Web applications have become fundamental to modern organizational operations, supporting services that depend on the confidentiality, integrity, and availability (CIA) of digital information [1], [2]. As organizations increasingly rely on web-based systems, the attack surface expands due to insecure configurations, outdated software components, weak authentication mechanisms, unsafe data handling, and insufficient monitoring [3], [4], [5], [6], [7]. Consequently, organizations require systematic and repeatable security assessment methods to identify vulnerabilities before they are exploited.

Correspondence Author: ✉rizqullahraihan@student.telkomuniversity.ac.id
To cite this article: Raihan, R., Irsan, M., & Wibowo, Y. F. A. (2026). Security Vulnerability Analysis of an Employee Attendance Website Using Black-Box Penetration Testing and OWASP Top 10 2025. *Journal of Deep Learning, Computer Vision and Digital Image Processing*, *4*(3), 183–195. https://doi.org/10.61255/decoding.v4i3.1724

This is an open access article under the CC BY-SA license

Vulnerability assessment (VA) and penetration testing (pentest) are complementary approaches. VA is useful for discovering known weaknesses, exposed services, configuration problems, and potentially vulnerable software versions. Pentesting adds contextual verification by testing how an application behaves when it receives unexpected input, unauthorized requests, repeated authentication attempts, or other controlled attack scenarios. Automated findings should not be treated as final proof because scanners may produce false positives or lack application context. Manual validation is therefore necessary to determine whether a finding is observable, exploitable, or operationally relevant [8], [9].

The research object is the employee-attendance website of PT. Jaka Kuasa Nusantara. The application supports authentication, dashboard access, attendance functions, attendance history, user-profile functions, and face-capture functionality. The presence of account and attendance information makes the protection of communication channels, session cookies, authentication controls, and server configuration important. The system had not previously undergone a comprehensive security assessment combining automated scanning and manual verification within an OWASP-oriented framework.

Previous security studies have demonstrated the usefulness of OWASP-based testing for identifying common web-application risks [10]. However, many assessments emphasize either scanner output or a limited set of manual attack scenarios. The gap addressed in this study is the need for an applied assessment that combines reconnaissance, automated VA, controlled manual testing, risk interpretation, and a practical mitigation roadmap for an operational employee-attendance website. The study also distinguishes between confirmed weaknesses, controls that performed correctly in the tested scenario, and areas that require internal verification [11], [12].

The objective of this study is to identify security weaknesses in the employee attendance website of PT. Jaka Kuasa Nusantara through vulnerability assessment and black-box penetration testing based on the OWASP Top 10 2025 framework, evaluate the effectiveness of selected security controls, and provide prioritized mitigation recommendations. This work contributes a practical assessment workflow suitable for organizational web security evaluation.

METHOD

Research Design

This study used an applied, qualitative, black-box security-assessment design [13], [14]. An authorized external tester evaluated the application without source-code access. The assessment combined observation, automated scanning, and controlled manual testing. The confidentiality, integrity, and availability (CIA) triad guided impact interpretation, while CVSS v3.1 concepts informed qualitative risk assessment [15].

Research Object and Authorized Scope

The target was the PT. Jaka Kuasa Nusantara employee-attendance website, including authentication, attendance recording and history, user profiles, and face-capture functions. Authorization and scope were agreed through communication with a company representative via WhatsApp. Testing excluded destructive exploitation, denial-of-service activity, data modification, and actions that could disrupt production services. No source code or internal server configuration was inspected.

Pre-engagement and Data Collection

Pre-engagement communication established the testing scope. Direct observation, vulnerability scanning, and controlled penetration-testing scenarios were then used to characterize externally observable application behavior. Sensitive infrastructure identifiers are omitted from the reporting.

Tools and Instrumentation

WHOIS supported domain reconnaissance, and Zenmap/Nmap identified reachable services [16]. Nessus identified potential configuration and component weaknesses. Kali Linux supported manual

testing, while browser developer tools were used to inspect responses, headers, cookies, redirects, and client-side resources.

Testing Procedure

The workflow comprised scope confirmation, reconnaissance, automated vulnerability analysis, non-destructive manual verification, impact assessment, and reporting [8], [13]. OWASP Top 10 2025 provided the risk categories used to organize findings [17]. A weakness was labeled identified when supported by scanner evidence or manual observation; not observed when the selected test did not demonstrate it; and requiring internal verification when external testing could not establish the presence or effectiveness of a server-side control.

Analysis

Findings were interpreted using the available evidence, potential CIA effects, exposure, and remediation priority. Scanner severity was not treated as conclusive evidence of exploitability. Qualitative priorities were distinguished from calculated CVSS scores, and negative test results were limited to the scenarios examined [15].

RESULTS AND DISCUSSION

Automated Vulnerability Assessment Results

Automated assessment identified concerns involving HTTP configuration, cookie attributes, response content types, and the reported server version. Table 1 summarizes the observations and their potential security implications. Severity is described qualitatively because verified finding-specific CVSS scores were not available for all items.

Table 1. Automated Vulnerability Assessment Results

Finding	Observed evidence	Potential security effect	Interpretation
HTTP Multiple Issues	HTTP configuration was identified as not fully optimized	May permit insecure communication or inconsistent transport protection	Confirmed configuration concern
Cookie without HttpOnly	A session/application cookie lacked the HttpOnly attribute	Client-side scripts may access the cookie if an XSS condition exists	Confirmed configuration concern
Cookie without Secure	A cookie lacked the Secure attribute	The cookie may be transmitted over an insecure channel if HTTP remains available	Confirmed configuration concern
Improper/missing Content-Type	Response content type was not configured as expected	May increase MIME interpretation or content-handling risk	Confirmed configuration concern
Apache 2.4.29	Apache 2.4.29 was detected; patch status was not verified	Known weaknesses may remain unpatched depending on build and backports	Requires version and patch verification

Manual OWASP-Oriented Testing Results

Manual testing was used to validate application behavior beyond scanner output. The tested scenarios did not demonstrate every possible OWASP weakness; therefore, the results are intentionally scoped. “Not observed” means that the selected test did not produce evidence of the weakness and must not be interpreted as a guarantee that the entire application is secure.

Table 2. Manual OWASP-Oriented Testing Results

OWASP-related area	Test scenario	Observed result	Status
A01:2025 – Broken Access Control	Attempted direct navigation to restricted paths and unauthorized resources and SSRF-related request patterns within the authorized scope	Restricted paths were rejected or returned a not-found response in the tested cases. No SSRF behavior was demonstrated in the tested functionality.	Not observed in tested scope
A02:2025 – Security Misconfiguration	Inspected headers, cookies, HTTP behavior, and server exposure	Missing/insecure cookie attributes and HTTP/header issues were identified	Identified
A03:2025 – Software Supply Chain Failures	Reviewed scanner evidence of server component version	Apache 2.4.29 required patch and backport verification	Identified/verification required
A04:2025 – Cryptographic Failures	Compared HTTP/HTTPS behavior and inspected transport-related responses	HTTPS was available; transport hardening and consistent HTTPS enforcement required improvement	Identified improvement area
A05:2025 – Injection	Submitted unexpected and injection-oriented input to the login process	No database error, authentication bypass, or abnormal response was observed	Not observed in tested scenario
A06:2025 – Insecure Design	Reviewed protective controls around automated authentication attempts	CAPTCHA and login-attempt limiting were observed	Control observed; continue hardening
A07:2025 – Authentication Failures	Repeated invalid login attempts were performed in a controlled manner	CAPTCHA and attempt-limiting controls were present	Control observed
A08:2025 – Software or Data Integrity Failures	Reviewed observed client-side resource loading	No direct evidence of suspicious external script loading was observed	Not observed in tested scope
A09:2025 – Security Logging and Alerting Failures	Observed application feedback during failed authentication	External testing could not confirm centralized logging or real-time alerting	Requires internal verification
A10:2025 – Mishandling of Exceptional Conditions	Submitted invalid/empty inputs and observed error handling	No detailed stack trace or database error was exposed	Not observed in tested scope

Risk Interpretation Using the CIA Triad

Table 3. Risk Interpretation Using the CIA Triad

Area	Confidentiality	Integrity	Availability	Overall priority
HTTP/HTTPS and cookie configuration	High: credentials/session data may be exposed under unsafe conditions	Medium: session misuse may affect account actions	Low–Medium	High
Security headers and content handling	Medium	Medium	Low	High

Area	Confidentiality	Integrity	Availability	Overall priority
Apache component requiring verification	Potentially high, depending on patch status	Potentially high	Potentially medium	High until verified
Authentication controls	Medium; account compromise risk is reduced by existing controls	Medium	Low	Medium
Logging and alerting	Indirect impact through delayed detection	Indirect impact through delayed response	Indirect impact	Medium–High

Discussion

The results indicate that the application already implements several useful defensive controls. The presence of CAPTCHA and login-attempt limiting reduces the practicality of automated credential attacks, while the absence of database error disclosure in the tested injection scenario limits information leakage. The rejection of tested unauthorized paths also suggests that some access-control boundaries are functioning. These positive observations should be retained and validated after future application changes.

Cookies without HttpOnly and Secure attributes weaken session protection. HttpOnly prevents client-side scripts from reading a cookie, while Secure restricts its transmission to HTTPS. These attributes provide defense in depth but do not replace controls against cross-site scripting or transport attacks. An appropriate SameSite policy and consistent configuration of session cookies should also be reviewed [18].

HTTP configuration issues should be addressed through a complete HTTPS-only policy [19]. All HTTP requests should be redirected permanently to HTTPS, and HTTP Strict Transport Security (HSTS) should be enabled after verifying that every required subdomain and service supports HTTPS. Modern TLS settings, secure certificate management, and the removal of mixed content should be reviewed together. Because the application includes login and attendance functions, transport protection is a high-priority control [20], [21].

The detected Apache 2.4.29 version should not be interpreted automatically as an exploitable vulnerability because some operating-system vendors backport security patches without changing the visible upstream version. The system owner should therefore verify the operating-system package version, vendor security advisories, installed patches, and actual build information [22]. Nevertheless, the finding is operationally important because unsupported or outdated components increase maintenance risk and may expose the system to publicly known vulnerabilities [23].

The security-header findings support a server-hardening program. The application should evaluate Content-Security-Policy, X-Content-Type-Options: nosniff, frame-embedding protection through CSP frame-ancestors or X-Frame-Options, Referrer-Policy, and Permissions-Policy [18]. These headers must be tested against application functionality because an overly restrictive policy can break legitimate scripts, styles, frames, or integrations [24].

The assessment also highlights the limits of external testing. Logging and alerting cannot be confirmed reliably from user-facing responses. The organization should inspect server logs, application logs, authentication events, retention settings, access controls, and alert rules. Centralized monitoring should detect repeated failed logins, unusual session behavior, privilege changes, configuration changes, and abnormal request patterns [25]. Logs should contain sufficient context for investigation while avoiding unnecessary storage of sensitive personal or biometric information [9].

Prioritized Mitigation Roadmap

Table 4. Prioritized Mitigation Roadmap

Priority	Action	Implementation guidance	Expected outcome
1	Enforce HTTPS and enable HSTS	Redirect all HTTP traffic to HTTPS; verify certificates, TLS configuration, and subdomain readiness before enabling HSTS	Reduces transport downgrade and credential/session exposure
2	Secure all cookies	Set Secure, HttpOnly, and an appropriate SameSite attribute; review cookie scope, lifetime, and renewal	Strengthens session confidentiality and integrity
3	Deploy security headers	Implement CSP, X-Content-Type-Options, frame-ancestors/X-Frame-Options, Referrer-Policy, and Permissions-Policy; test in report-only mode where appropriate	Reduces browser-side attack surface
4	Patch and inventory components	Create a software inventory; verify Apache package/backport status; update server, framework, libraries, and OS components	Reduces exposure to known vulnerabilities
5	Strengthen authentication	Retain CAPTCHA and attempt limiting; add rate limiting, strong password policy, MFA/2FA, secure session rotation, and timeout	Reduces account compromise risk
6	Centralize logging and alerting	Collect application, web-server, authentication, and system logs; define alerts and retention rules	Improves detection and incident response
7	Perform periodic security testing	Repeat VA/pentest after major changes and at scheduled intervals; track remediation evidence	Maintains security posture over time

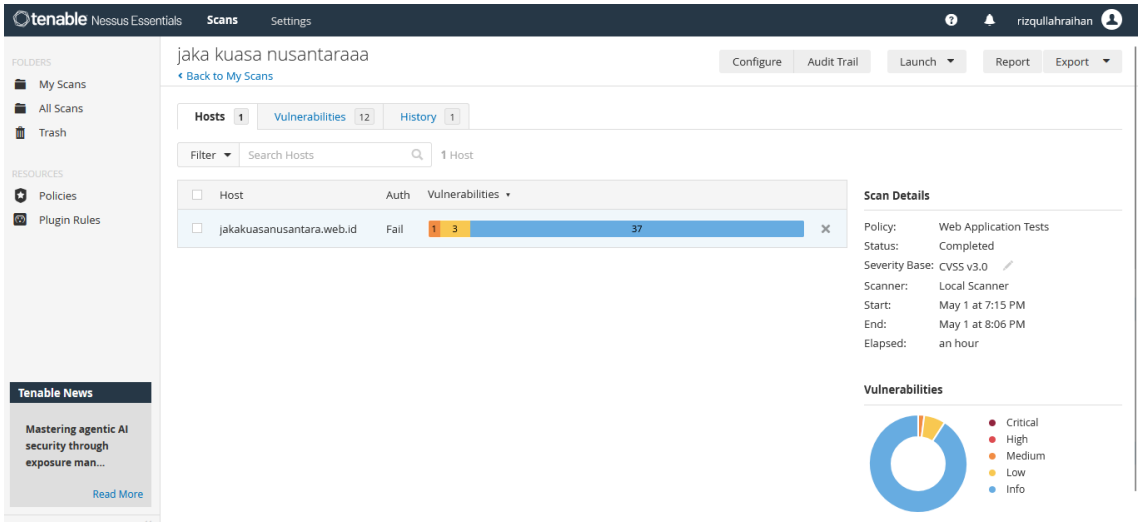


Figure 1. Nessus vulnerability assessment dashboard and scan summary

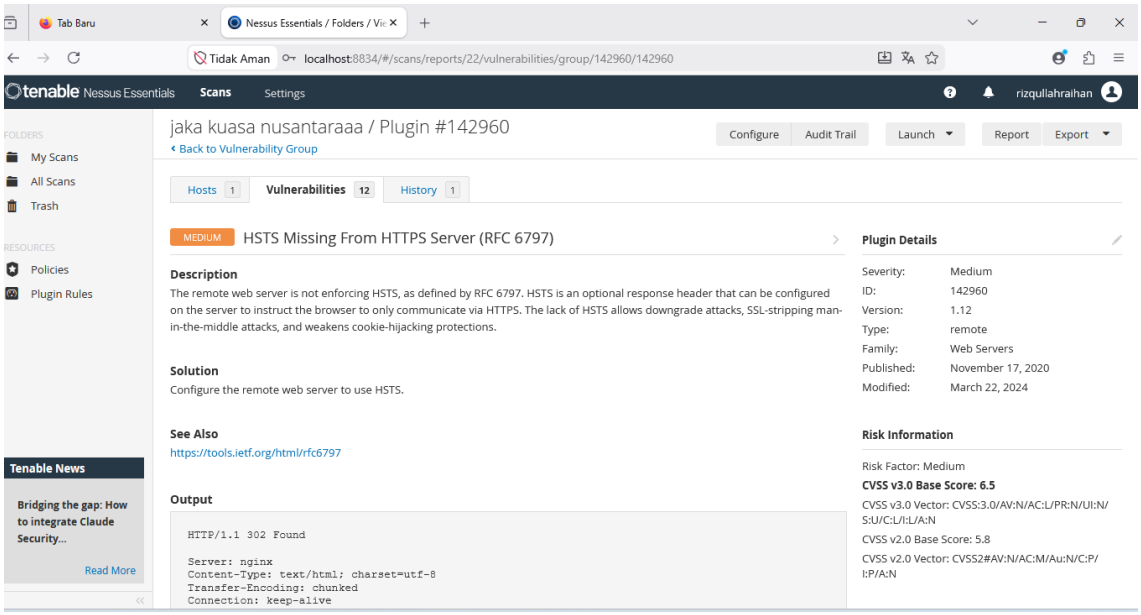


Figure 2. Detailed Nessus finding used as evidence for configuration analysis

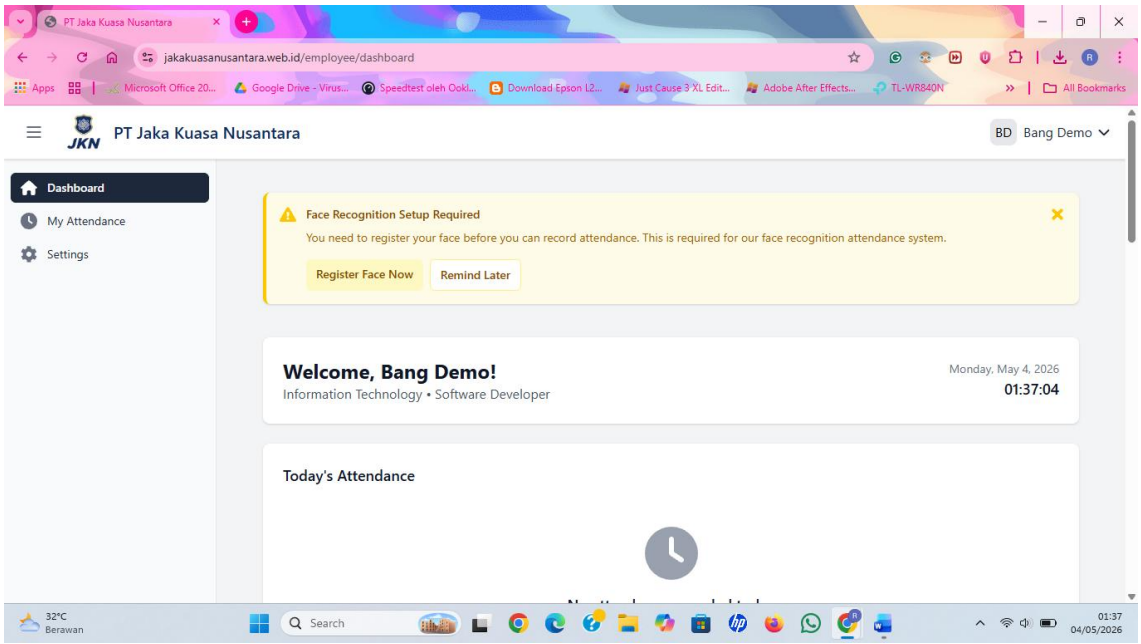


Figure 3. Login testing scenario showing the application's authentication protection behavior

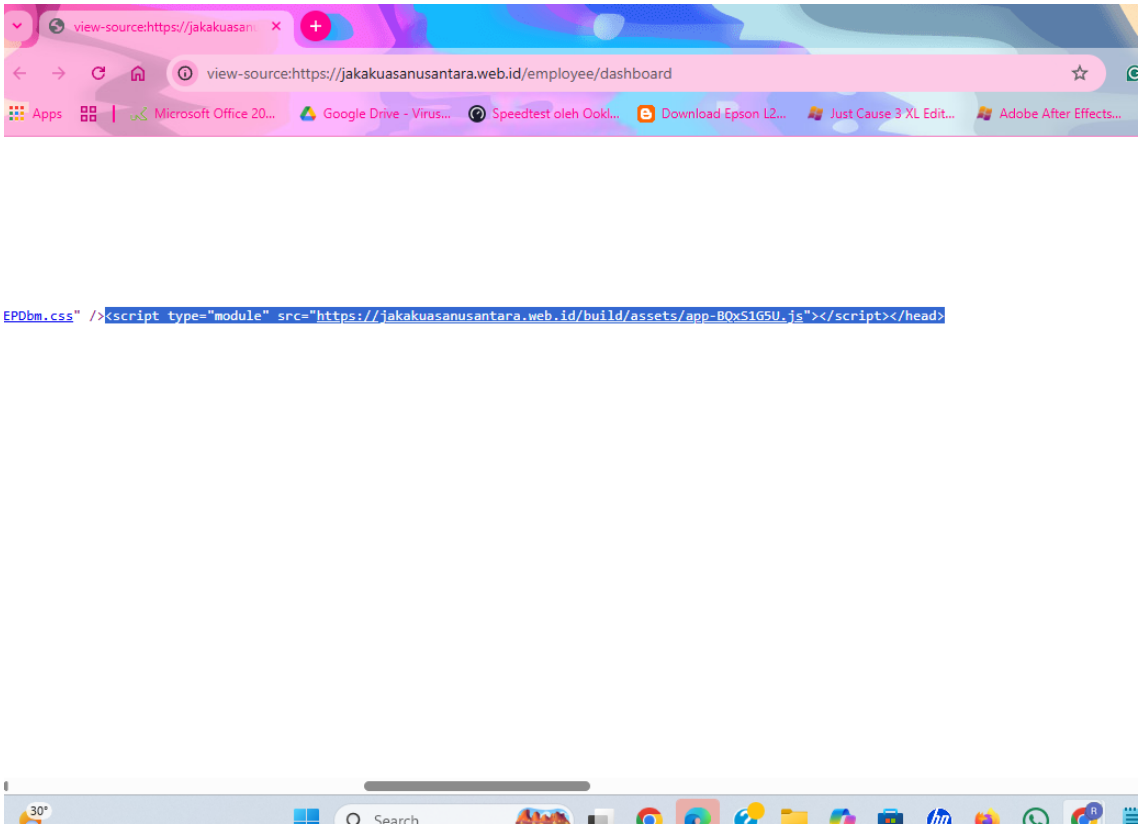


Figure 4. Browser developer-tools evidence used to inspect HTTP responses and security-related configuration

Figures 1–4 provide examples of the automated findings, authentication behavior, and browser-based inspection used in the assessment.

Evidence from Manual Testing and Mitigation Traceability

Figures 5–9 document selected manual tests. Their interpretation is limited to the endpoints, roles, inputs, and responses examined during the authorized assessment.

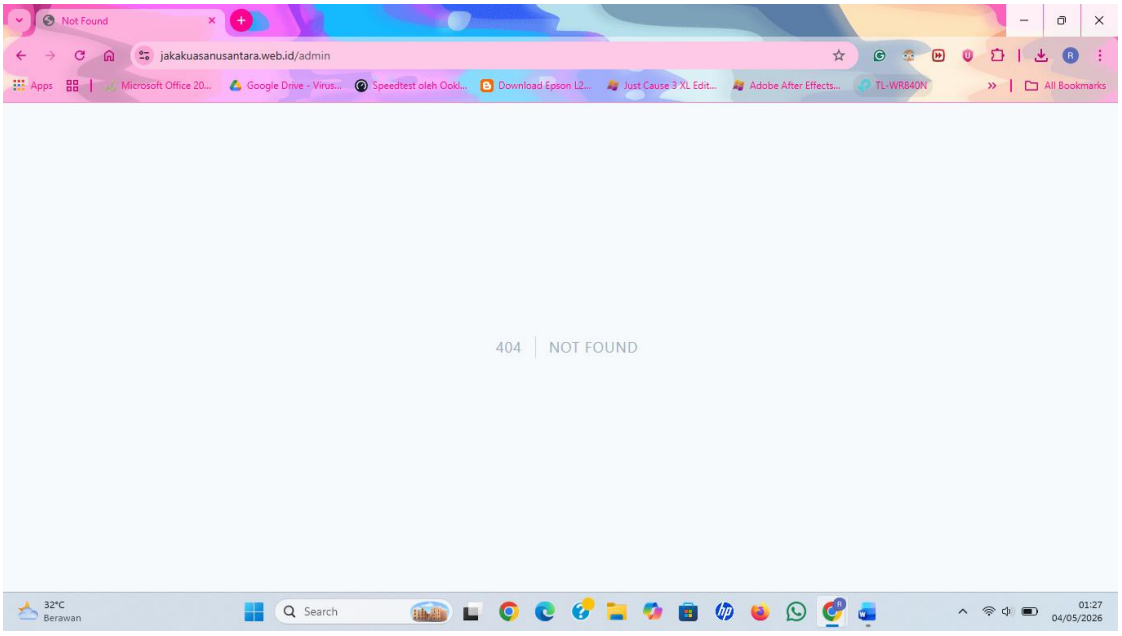


Figure 5. Direct-access test of the administrative path. The application redirected the unauthenticated request to the login interface

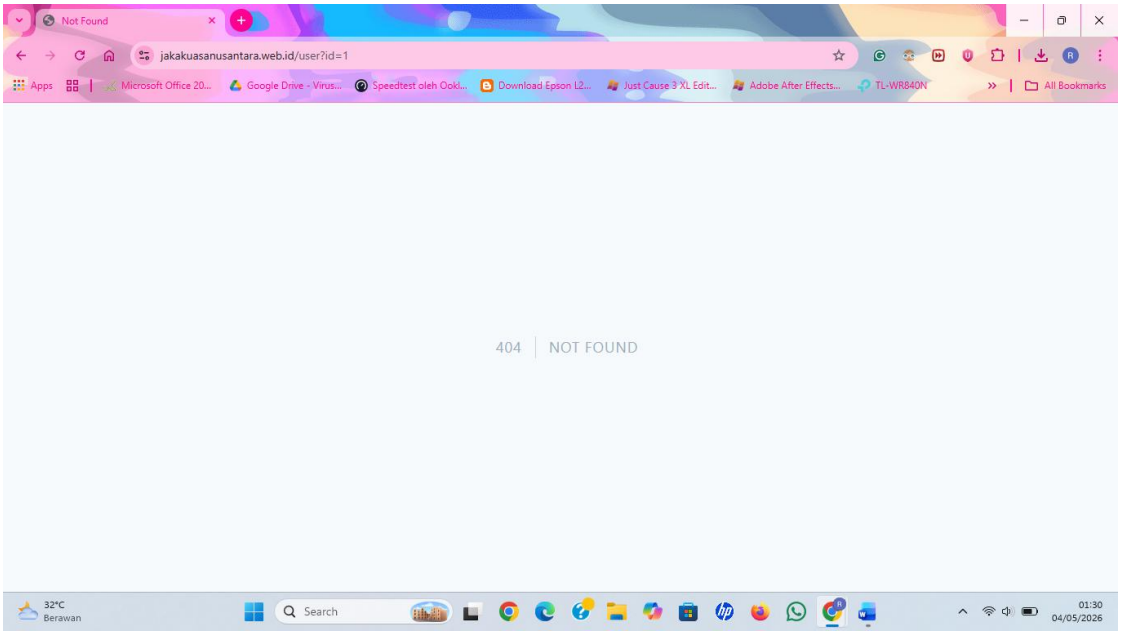


Figure 6. Direct-object access test using another user identifier. The tested request did not provide access to another user's data

The access-control tests support the conclusion that no Broken Access Control weakness was demonstrated in the selected scenarios. The administrative path required authentication, while attempts to access another user's identifier did not expose the requested user data. These results are limited to the tested URLs and roles; comprehensive authorization testing should include all user roles, endpoints, APIs, and state-changing functions.

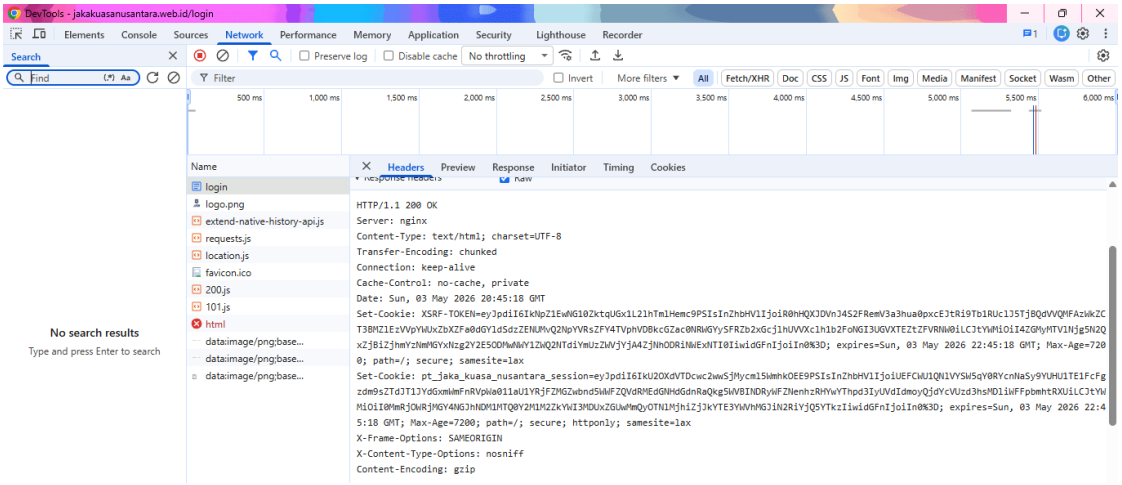


Figure 7. Browser developer-tools inspection used to examine HTTP response headers and security-related configuration

The inspected response did not include Strict-Transport-Security or Content-Security-Policy, supporting the reported security-configuration concerns. Recommended hardening includes an appropriate Content-Security-Policy, frame-embedding restrictions, and X-Content-Type-Options. Each configuration should be tested against the application's legitimate functionality [18].

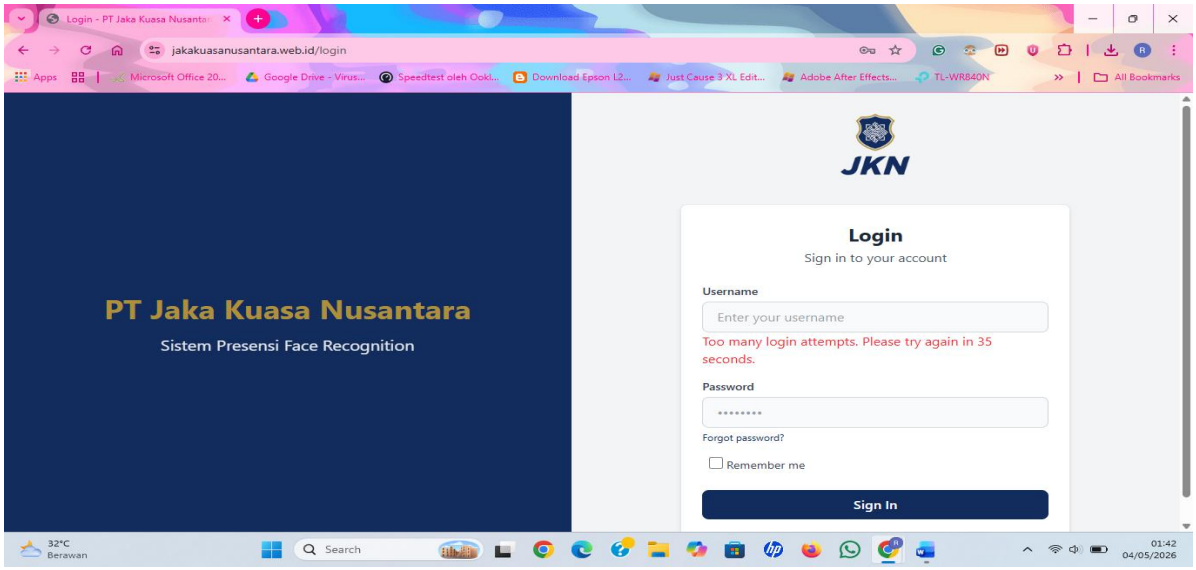


Figure 8. Authentication test showing the login-attempt cooldown observed after repeated invalid password attempts

The application imposed a cooldown after the sixth failed login attempt (Figure 8). This behavior demonstrates an attempt-limiting control in the tested scenario. It does not establish the effectiveness of authentication across distributed attacks, alternative endpoints, or other account workflows.

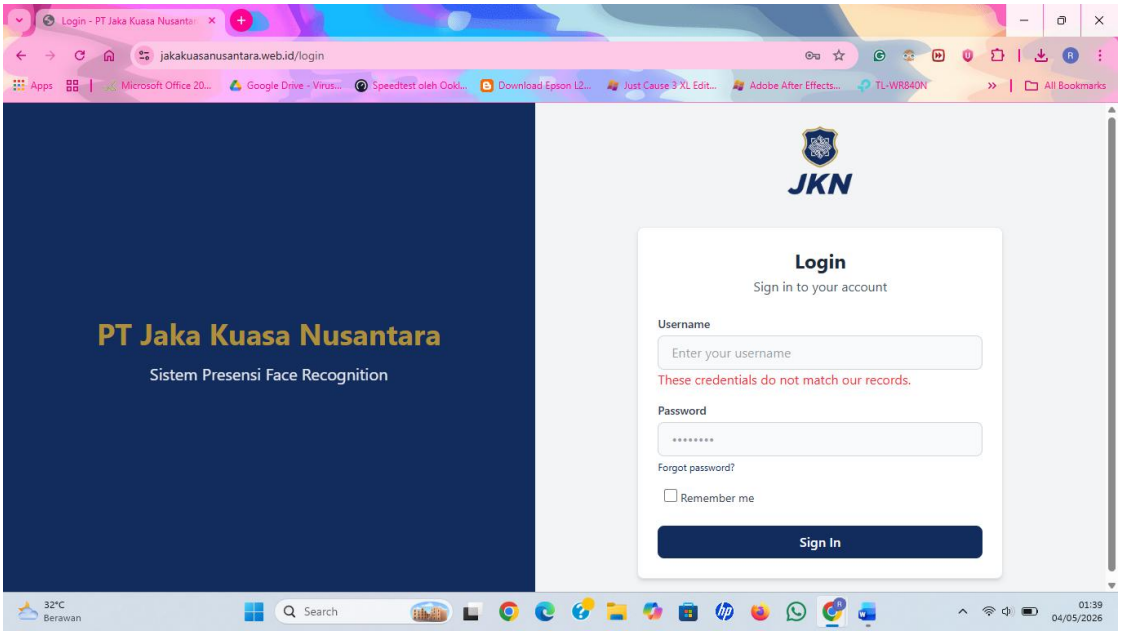


Figure 9. Controlled invalid-login scenario used to observe authentication feedback and protective behavior

The authentication assessment observed CAPTCHA and a login-attempt limitation. These controls provide a defensive layer against automated credential attacks. The scope also recommends stronger password requirements, secure session handling, session timeout, session-ID regeneration after login, complete session invalidation at logout, and monitoring of unusual login activity. These recommendations are presented as mitigation actions rather than as newly discovered vulnerabilities.

Research Contribution, Limitations, and Future Work

This study combines automated vulnerability assessment with controlled manual verification and maps the observations to OWASP Top 10 2025 [17]. The workflow distinguishes observed weaknesses, controls that operated as expected, and conditions requiring internal verification. The assessment was limited to selected externally observable scenarios and excluded source-code review, destructive exploitation, and comprehensive internal-infrastructure testing. Verified CVSS scores were not available for every scanner finding, so qualitative remediation priorities were used. Future work should include role-based authorization testing, API assessment, source-code and dependency review, server-configuration review, log analysis, and retesting after remediation.

CONCLUSION

This study assessed the PT. Jaka Kuasa Nusantara employee-attendance website using an authorized black-box approach that combined WHOIS, Zenmap/Nmap, Nessus, Kali Linux, browser-based inspection, and OWASP Top 10 2025-oriented manual testing. The assessment identified HTTP configuration issues, cookies without HttpOnly and Secure attributes, an improper or missing Content-Type configuration, and an Apache 2.4.29 component that requires patch-status verification. Manual testing did not demonstrate SQL injection or SSRF in the tested scenarios, while CAPTCHA and login-attempt limiting were observed as positive authentication controls. The results show that the application has an existing security foundation but requires systematic hardening, particularly for transport security, cookie protection, security headers, component maintenance, and operational monitoring. The highest-priority actions are enforcing HTTPS and HSTS, securing cookies, deploying and testing appropriate security headers, verifying and patching software components, strengthening authentication with layered controls, centralizing logging and alerting, and conducting periodic reassessment. Because the study was limited to externally observable behavior, the findings should be treated as an evidence-based security baseline rather than a complete certification of security.

ACKNOWLEDGMENT

The authors thank PT. Jaka Kuasa Nusantara for authorizing access within the agreed research scope and the academic supervisors and Faculty of Informatics at Telkom University for their guidance and support.

AUTHOR CONTRIBUTION STATEMENT

RR: conceptualization, methodology, literature review, data collection, vulnerability assessment, manual testing, analysis, visualization, writing—original draft, and writing—review and editing.

AI DISCLOSURE STATEMENT

Claude.ai was used during manuscript preparation for language refinement, grammar checking, sentence restructuring, and academic clarity. ChatGPT (OpenAI) was used during copyediting for language, formatting, and consistency checks. The authors remain responsible for the scientific content and the final manuscript.

REFERENCES

[1] A. A. Adebayo, S. Misra, and R. Maskeliūnas, "Cybersecurity in the era of digital transformation: A review," *Journal of Information Security and Applications*, vol. 58, Art. no. 102804, 2021, doi: 10.1016/j.jisa.2021.102804.

[2] ISO/IEC, ISO/IEC 27001:2022 Information Security, Cybersecurity and Privacy Protection—Information Security Management Systems—Requirements. Geneva, Switzerland: International Organization for Standardization, 2022.

[3] H. Kayan, M. Nunes, O. Rana, P. Burnap, and C. Perera, "Cybersecurity of industrial cyber-physical systems: A review," *ACM Computing Surveys*, vol. 55, no. 1, 2022, doi: 10.1145/3510410.

[4] A. Ahmad, R. Maulana, and M. Yassir, "Cybersecurity challenges in the era of digital transformation: A comprehensive analysis of information systems," *Journal Informatic, Education and Management*, vol. 6, no. 1, pp. 7–11, 2024, doi: 10.61992/jiem.v6i1.57.

[5] G. Dhillon, "Security of the digital transformation," *Computers & Security*, vol. 108, Art. no. 102345, 2021, doi: 10.1016/j.cose.2021.102345.

[6] A. Hanelt, R. Bohnsack, D. Marz, and C. A. Marante, "A systematic review of the literature on digital transformation: Insights and implications for strategy and organizational change," *Journal of Management Studies*, vol. 58, no. 5, pp. 1159–1197, 2021, doi: 10.1111/joms.12639.

[7] M. Haleem, M. Javaid, R. P. Singh, S. Rab, and R. Suman, "Perspectives of cybersecurity for Industry 4.0: A review-based framework," *Industrial Robot*, vol. 49, no. 3, pp. 582–597, 2022, doi: 10.1108/IR-10-2021-0243.

[8] OWASP Foundation, "OWASP web security testing guide," 2024. [Online]. Available: https://owasp.org/www-project-web-security-testing-guide/.

[9] National Institute of Standards and Technology*, Guide for Conducting Risk Assessments*, NIST Special Publication 800-30 Rev. 1, 2012, doi: 10.6028/NIST.SP.800-30r1.

[10] M. R. Abdallah, P. Hatta, and C. W. Budiyanto, "Security analysis of web-based information systems through vulnerability assessment using the framework of OWASP web security testing guide and common vulnerability scoring system," *Journal of Informatics and Vocational Education*, vol. 7, no. 3, 2024, doi: 10.20961/joive.v7i3.2411.

[11] C. Darmawan, J. P. P. Naibaho, and A. De Kweldju, "Penerapan metode vulnerability assessment untuk identifikasi keamanan website berdasarkan OWASP Top 10 tahun 2021," *Edumatic: Jurnal Pendidikan Informatika*, vol. 8, no. 1, 2024, doi: 10.29408/edumatic.v8i1.25834.

[12] S. E. Sadat, M. F. Naseri, and K. Salamzada, "Identifying and mitigating web application vulnerabilities: A comparative study of countermeasures and tools," *International Journal of Software Engineering and Computer Science*, vol. 4, no. 3, 2024, doi: 10.35870/ijsecs.v4i3.3138.

[13] National Institute of Standards and Technology, *Technical Guide to Information Security Testing and Assessment*, NIST Special Publication 800-115, 2008, doi: 10.6028/NIST.SP.800-115.

[14] R. S. Pressman and B. R. Maxim, *Software engineering: A practitioner's approach*, 9th ed. New York, NY, USA: McGraw-Hill Education, 2019.

[15] FIRST, "Common vulnerability scoring system version 3.1: Specification document," 2019. [Online]. Available: https://www.first.org/cvss/.

[16] G. F. Lyon, *Nmap network scanning*. Sunnyvale, CA, USA: Insecure.Org, 2009.

[17] OWASP Foundation, "OWASP Top 10:2025," 2025. [Online]. Available: https://top10.owasp.org/2025/.

[18] OWASP Foundation, "OWASP cheat sheet series," 2025. [Online]. Available: https://cheatsheetseries.owasp.org/.

[19] R. Fielding et al., "*HTTP semantics*," RFC 9110, Jun. 2022, doi: 10.17487/RFC9110.

[20] E. Rescorla, "*The transport layer security (TLS) protocol version 1.3*," RFC 8446, Aug. 2018, doi: 10.17487/RFC8446.

[21] J. Hodges, C. Jackson, and A. Barth, "*HTTP strict transport security (HSTS)*," RFC 6797, Nov. 2012, doi: 10.17487/RFC6797.

[22] Apache Software Foundation, "Apache HTTP server documentation," 2025. [Online]. Available: https://httpd.apache.org/docs/.

[23] Cybersecurity and Infrastructure Security Agency, "Known exploited vulnerabilities catalog," 2025. [Online]. Available: https://www.cisa.gov/known-exploited-vulnerabilities-catalog.

[24] Tenable, "Nessus documentation," 2025. [Online]. Available: https://docs.tenable.com/nessus/.

[25] B. Uchendu, J. R. C. Nurse, M. Bada, and S. Furnell, "Developing a cyber security culture: Current practices and future needs," *Computers & Security*, vol. 109, Art. no. 102387, 2021, doi: 10.1016/j.cose.2021.102387.

[26] I. Lee, "Cybersecurity: Risk management framework and investment cost analysis," *Business Horizons*, vol. 64, no. 5, pp. 659–671, 2021, doi: 10.1016/j.bushor.2021.02.022.