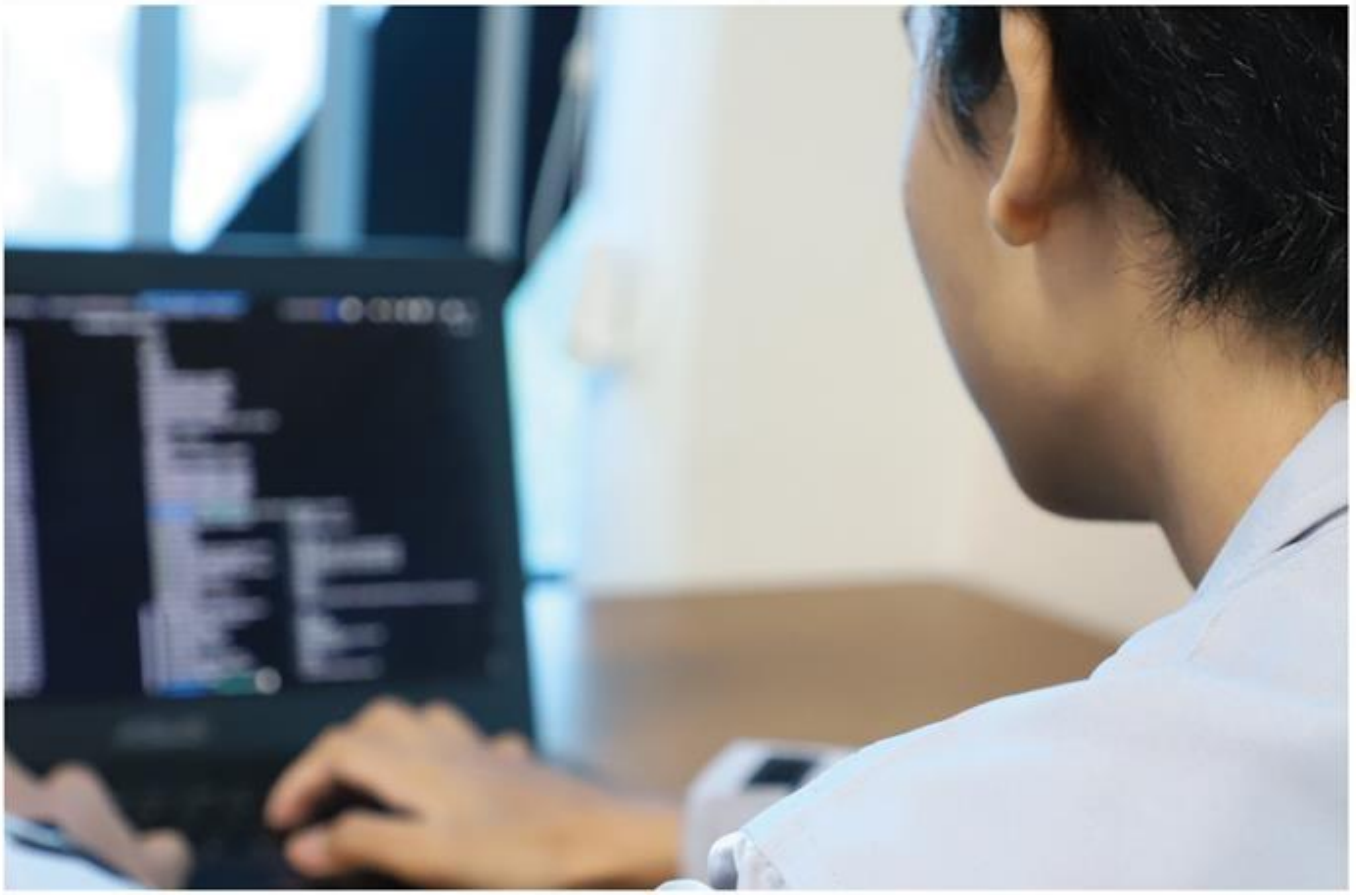
FINAL REPORT

PROJECT BASED LEARNING

“VULNERABILITY ASSESSMENT DAN PENETRATION TESTING 5”



CYBER SECURITY ENGINEERING STUDY PROGRAM

DEPARTMENT OF INFORMATION TECHNOLOGY

BATAM STATE POLYTECHNIC

2026

Project Identity

Project Title : Vulnerability Assessment and Penetration Testing 5

Project Proposer : Maidel Fani, S.Pd., M.Kom.

Project Manager : Agus Fatulloh, S.T., M.T

Client : STAI Paduka Anambas

Output : 1. Final Report
2. Penetration Testing Report
3. Poster

PBL Members :

NO	Name	NIM	Task Report	Jobdesk Description
1.	Inanta Rahmatan Ilahi (Leader)	4332301014	Con and Suggestion, Appendix, and Document Editing	1. Conduct passive reconnaissance 2. Identify assets 3. Perform analysis using the Static Application Security Testing (SAST) method 4. WSTG-SESS 01-10 and WSTG-INPV 11-20 5. Exploiting SQL Injection attacks (Time-Based & Code Logic) and Open Redirection (URL Redirection) 6. Creating a final report on the PTES method and creating a bast file
2.	Balqis Juwita Candra	4332301002	Methodology	1. Conduct passive reconnaissance 2. Identify assets 3. Perform analysis using the Static Application Security Testing (SAST) method 4. WSTG INFO 01-10 and WSTG-CLNT 01-14 5. Exploit Local Denial of Service (DoS) and Cross-Site Scripting (XSS) attacks 6. Create a final PowerPoint presentation

3.	Cecilia Revi Marischa Putri	4332301019	Result and Discussion	1. Conduct passive reconnaissance 2. Identify entry points 3. Perform analysis using the Dynamic Application Security Testing (DAST) method 4. WSTG-IDNT 01-05, WSTG-ATHZ 01-05, and WSTG-BUSL 01-10 5. Exploiting Brute Force attacks, Broken Access Control (Access Handling Errors), and Information Disclosure (Directory Browsing) 6. Creating a final penetration testing report and creating a poster
4.	Mariza Hariyanti	4332301023	Background and References	1. Conduct active reconnaissance 2. Identify threat actors 3. Perform analysis using the Dynamic Application Security Testing (DAST) method 4. WSTG-CONF 01-13, WSTG-ERRH 01-02, WSTG-CRYP 01-04, and WSTG-APIT 01 5. Exploiting Data Integrity Violation (Data Storage Failure) and Cross-Site Request Forgery (CSRF) attacks 6. Creating a manual book
5.	Richky Nugraha Suhandinata Simanjuntak	4332301016	Theoretical Based	1. Conduct active reconnaissance 2. Identify vulnerabilities 3. Map attacks prior to exploitation 4. WSTG-ATHN 01-11 and WSTG-INPV 01-10 5. Exploit Prototype Pollution (Vulnerable JS Library) and Clickjacking attacks 6. Create a demo video

Forewords

All praise and gratitude are extended to God Almighty for His abundant blessings and grace, which have enabled the researchers to successfully prepare and complete this Project Based Learning (PBL) Report for the fifth semester, entitled **"Vulnerability Assessment and Penetration Testing 5."** This report is compiled as one of the mandatory academic requirements to be eligible for the Final Semester Examination of the Cyber Security Engineering Study Program at Politeknik Negeri Batam. This report is developed based on a project proposed by the client, **Damar Aqsa, A.Md.Kom**, who has entrusted the researchers with the opportunity to work on a real-world, practical, and industry-relevant cybersecurity project. Through this project, the researchers were able to apply theoretical knowledge and technical skills acquired during the learning process into an actual implementation that reflects current cybersecurity challenges and practices. Throughout the process of conducting this project and preparing the report, the researchers acknowledge that its successful completion would not have been possible without the valuable support, guidance, and contributions from various parties. Therefore, the researchers would like to express their sincere appreciation and gratitude to:

1. **Mr. Agus Fatulloh, S.T., M.T.**, as the Project Based Learning Manager, for his supervision, direction, and support throughout the implementation of this project.
2. **All lecturers of the Cyber Security Engineering Study Program at Politeknik Negeri Batam**, for their continuous guidance, knowledge sharing, and academic direction during the learning process and the completion of this report.
3. **Damar Aqsa, A.Md.Kom**, as the client, for the trust, insights, guidance, and support provided during the execution of this project.
4. **All group members**, for their strong collaboration, dedication, and commitment, which enabled the project to be completed effectively and optimally.

The researchers fully realize that this report still has limitations due to constraints in knowledge, experience, and time. Therefore, constructive criticism and suggestions are sincerely expected to improve the quality of this report in the future. Finally, it is hoped that this report will provide meaningful benefits, not only for the researchers as part of their academic and professional development, but also for readers, practitioners, and other parties interested in the field of cybersecurity, particularly in vulnerability assessment and penetration testing.

Batam, January 9, 2026

Table of Content

Table of Contents

List Of Images

1. Background

The advancement of information technology has significantly increased the use of web applications across various sectors, including the management and publication of scientific journals. Web-based journal platforms not only serve as a medium for disseminating knowledge but also function as systems that manage important and sensitive data, such as user identities, research manuscripts, and editorial processes. Therefore, web application security is a critical aspect in maintaining service reliability, user trust, and the integrity of academic data. xxx.com is an academic journal platform that utilizes the Open Journal System (OJS) as its publication management system. OJS provides essential features such as user authentication, manuscript submission and management, role-based access control, and editorial workflow support. The complexity of these features may introduce security vulnerabilities if they are not supported by secure system configurations and adequate security testing mechanisms. Weaknesses within the system can be exploited for account misuse, journal data manipulation, information leakage, or disruption of publication services.

As a publicly accessible web application, OJS is exposed to various cyber threats, including SQL Injection, Cross-Site Scripting (XSS), Local File Inclusion (LFI), brute-force attacks, and insecure system configurations. These attacks pose serious risks to the confidentiality, integrity, and availability of the system. If not properly addressed, such security risks may negatively affect the credibility of the journal and compromise the continuity of the academic publication process. Based on these conditions, a structured and comprehensive security evaluation of xxx.com is required. Vulnerability Assessment and Penetration Testing (VAPT) is conducted to identify existing vulnerabilities, assess their risk levels, and evaluate their potential exploitability. This security assessment adopts the Penetration Testing Execution Standard (PTES) as the primary framework and the OWASP Web Security Testing Guide (WSTG) as the technical guideline for web application testing. Through this approach, the assessment is expected to produce accurate security findings and effective mitigation recommendations to improve the overall security posture of the system.

2. Theoretical Based

2.1 Vulnerability Assessment and Penetration Testing

Vulnerability Assessment and Penetration Testing (VAPT) is a systematic approach used to identify, analyze, and evaluate security weaknesses within information systems, applications, and network infrastructures. Vulnerability assessment focuses on the process of detecting potential vulnerabilities through automated scanning and manual analysis without actively exploiting the identified weaknesses. In contrast, penetration testing simulates real-world cyberattacks by exploiting validated vulnerabilities to assess the actual impact and risk posed to the system. This approach helps organizations understand how attackers may compromise confidentiality, integrity, and availability of information assets. By implementing VAPT, organizations can proactively strengthen their security posture, prioritize remediation efforts, and reduce the likelihood of successful cyberattacks.

2.2 Web Application Security Principles

Web application security is a critical component in ensuring the reliability and continuity of digital services, particularly for platforms that manage public information and sensitive data. Application security focuses on protecting three fundamental principles: confidentiality, integrity, and availability. Confidentiality ensures that data is accessible only to authorized parties, preventing information leakage or unauthorized disclosure. Integrity guarantees that data remains accurate, consistent, and protected from unauthorized modification during processing or storage. Availability ensures that systems and services remain accessible to legitimate users whenever needed without disruption. However, various security threats such as broken access control, insecure server configurations, and injection attacks can undermine these principles, potentially leading to data breaches, service disruption, and systemic abuse of the application.

2.3 White Box Testing Methodology

This project implements the White Box Testing approach, a testing methodology in which testers are granted full authority and access to the internal assets of the application. In this model, testers are allowed to review source code, use valid login credentials, and access architecture documentation as well as internal testing environments. Such access enables an in-depth analysis of application logic, security configurations, authentication mechanisms, and input validation processes. White Box Testing also supports the integration of static analysis and dynamic testing techniques to identify weaknesses at both the code and runtime levels. Through controlled exploitation, identified vulnerabilities can be validated accurately, allowing for precise assessment of security risks and their potential impact.

2.4 Penetration Testing Execution Standard (PTES)

PTES is a standard that provides a comprehensive workflow for conducting penetration testing in a systematic and well-structured manner. This standard ensures that each stage of the testing process, starting from pre-engagement interaction, information gathering, threat modeling, vulnerability analysis, exploitation, and final reporting, is carried out professionally in accordance with information security industry standards. By adopting PTES, penetration testing activities can be executed with clearly defined objectives, a well-scoped assessment, and controlled execution. The use of this standard also assists the researcher in maintaining methodological consistency and improving the reliability of security assessment results. As a result, the findings and recommendations generated through PTES-based testing can be effectively utilized to strengthen the overall security posture of the evaluated system.

2.5 OWASP Web Security Testing Guide (WSTG)

The OWASP Web Security Testing Guide (WSTG) is used as a technical guideline to assist in the vulnerability analysis process for web applications. This standard supports the Vulnerability Analysis stage in PTES by providing systematic and structured testing coverage of various aspects of application security. The testing categories used in OWASP WSTG include:

- **Information Gathering**: The initial phase aimed at identifying the technologies used, application structure, frameworks, endpoints, and exposed services, as well as mapping potential attack surfaces that could be targeted by attackers.
- **Configuration & Deployment Management Testing**: Evaluation of server, application, and deployment configurations to identify misconfigurations, default settings, or insecure services that could introduce security vulnerabilities or unauthorized access.
- **Identity Management Testing**: Testing mechanisms related to user identity lifecycle, including account creation, modification, and deletion, to ensure proper controls are in place to prevent identity misuse or privilege escalation.
- **Authentication Testing**: Assessment of authentication mechanisms such as login processes, password policies, and credential handling to ensure resistance against attacks like authentication bypass, brute-force attempts, and credential stuffing.
- **Authorization Testing**: Evaluation of access control mechanisms to ensure users can only access resources, functions, and data that align with their assigned roles and permissions.
- **Session Management Testing**: Analysis of session handling mechanisms, including session identifiers, cookies, tokens, and timeout policies, to prevent session hijacking, fixation, or unauthorized reuse.

- **Input Validation Testing**: Testing the application's ability to properly validate and sanitize user inputs to protect against malicious payloads and injection attacks such as SQL Injection and Cross-Site Scripting (XSS).
- **Error Handling Testing**: Evaluation of error handling and exception management mechanisms to ensure that application errors do not expose sensitive system information or internal implementation details.
- **Cryptography Testing**: Analysis of cryptographic implementations, including encryption algorithms, hashing mechanisms, and key management practices, to ensure sensitive data is securely stored and transmitted.
- **Business Logic Testing**: Testing the consistency and integrity of application workflows and business rules to detect logic flaws that could allow abuse of system functions or unauthorized actions.
- **Client-Side Testing**: Evaluation of security controls on the client side, such as JavaScript execution, browser storage, and user interactions, to identify vulnerabilities that could be exploited through the client environment.
- **API Testing**: Testing the security of API endpoints and web services to ensure proper authentication, authorization, input validation, and protection against common API-related attacks.

2.6 Security on the Open Journal System (OJS) Platform

The *xxx.com* platform operates using the Open Journal Systems (OJS) infrastructure. As a scientific publication platform, OJS presents specific attack vectors that may be exploited across several critical workflows. These include user authentication and editorial processes, which pose risks of account manipulation or unauthorized access to unpublished manuscripts. The manuscript upload functionality also represents a potential attack surface, particularly if file handling controls are insufficient, potentially enabling data exfiltration or the execution of malicious code. Additionally, system configuration weaknesses, such as Local File Inclusion (LFI), susceptibility to brute-force attacks, and insecure default configurations, can threaten the integrity and confidentiality of scientific manuscript data.

3. Methodology

Methodology is the main foundation in the implementation of Vulnerability Assessment and Penetration Testing (VAPT). This project applies a Hybrid-Structured Methodology approach that combines two global industry standards: the Penetration Testing Execution Standard (PTES) as a workflow framework, and the OWASP Web Security Testing Guide (WSTG) as a technical guide for in-depth testing. The testing workflow is divided into seven systematic phases aligned with PTES standards:

3.1 Pre-Engagement Interaction

This phase involves negotiations and the establishment of operational boundaries to ensure the legality and ethical compliance of the penetration testing activities. During this stage, the Rules of Engagement (RoE) are defined to determine technical parameters, testing schedules, and emergency contact procedures. The testing process is conducted using a dual-environment approach, where the Production environment is utilized to validate real system configurations under strict operational limitations, while the Localhost environment is used to simulate potentially destructive attacks without affecting service availability. In addition, legal aspects are addressed through the signing of a Non-Disclosure Agreement (NDA) to ensure the confidentiality and protection of sensitive client data throughout the testing process.

3.2 Intelligence Gathering

Although the White Box Testing approach was applied in this assessment, the Intelligence Gathering phase was still conducted to obtain a comprehensive understanding of the target system from an external perspective. In this phase, the researcher performed reconnaissance activities to map the attack surface by identifying subdomains, hosting infrastructure, and publicly exposed services related to the OJS platform. Technology fingerprinting was also carried out to determine the web server, backend programming language, database, framework versions, and third-party components in use, which are critical for identifying potential vulnerabilities. Additionally, SSL/TLS analysis was performed to evaluate the implementation of encryption mechanisms and the validity of security certificates protecting data in transit. The information collected during this phase served as foundational input for subsequent stages, particularly Threat Modeling and Vulnerability Analysis, enabling the testing process to be conducted in a more focused and effective manner.

3.3 Threat Modeling

The threat modeling stage began with the identification of critical assets divided into three categories: information assets, technology assets, and operational assets, which include highly sensitive data such as database credentials, SMTP settings, and the system's security salt.

Furthermore, the researcher identified various potential entry points from both external and internal actors, ranging from public login forms and file upload features to administrative dashboards and plugin management interfaces. Mapping these attack vectors revealed significant risks, including SQL Injection on search forms, Remote Code Execution (RCE) through malicious file uploads, and session hijacking resulting from weak session configurations. Detailed source-code analysis also uncovered critical weaknesses such as the use of the outdated SHA-1 hashing algorithm and empty API secret values, which could be exploited by threat actors. Ultimately, this stage provides a strategic foundation for prioritizing security remediation by understanding how different threat actors, from script kiddies to malicious insiders, might compromise the integrity and confidentiality of the OJS platform.

3.4 Vulnerability Analysis

The vulnerability analysis phase integrates the Penetration Testing Execution Standard (PTES) framework with the OWASP Web Security Testing Guide (WSTG) to systematically identify security gaps within the OJS platform. This process involves Static Application Security Testing (SAST) to audit the source code for insecure implementations, such as the use of the SHA-1 hashing algorithm, alongside Dynamic Application Security Testing (DAST) to detect runtime vulnerabilities including Cross-Site Scripting (XSS) and SQL Injection. Each identified vulnerability is classified according to the OWASP WSTG framework, covering critical domains such as Authentication, Authorization, and Input Validation to ensure a comprehensive evaluation. By correlating these findings, the researcher successfully identified high-risk configurations, including empty API secret values and weak session management mechanisms, which could be exploited by malicious actors.

3.5 Exploitation

The exploitation stage focuses on verifying the vulnerabilities identified during the analysis phase through controlled penetration attempts to demonstrate their real-world impact on the target system. In this project, the researcher conducted systematic exploitation activities on eleven distinct attack vectors, ranging from common techniques such as SQL Injection and Brute Force attacks to more advanced threats including Prototype Pollution and Clickjacking. Based on the results, seven vulnerabilities were successfully validated as exploitable security flaws that posed tangible risks to the application. One of the most critical findings involved a Time-Based SQL Injection vulnerability on the *csrfToken* parameter, which was exploited to infer and extract sensitive database information. In addition, a Brute Force vulnerability was successfully exploited due to the absence of effective rate-limiting and account lockout mechanisms, allowing repeated authentication attempts that could lead to unauthorized account access. Although not all exploitation attempts succeeded, each test was

documented to assess the application's security impact.

3.6 Post-Exploitation

The post-exploitation phase was conducted after all identified attacks from the previous phase were successfully validated within the authorized testing environment. At this stage, the focus of testing shifted from technical exploitation activities to analyzing the security impact resulting from each successfully executed attack across the seven successfully validated vulnerabilities. The analysis involved assessing the impact on the Confidentiality, Integrity, and Availability (CIA) aspects for each attack type, including Brute Force, SQL Injection, Clickjacking, and Open Redirection. The researcher determined the risk level of each vulnerability based on exploitation severity and potential business impact, categorizing the risks from medium to high. The results of this analysis are presented in a structured risk table that includes specific mitigation recommendations to support the prioritization of remediation efforts based on the severity and impact of each vulnerability on the security of the OJS platform.

3.7 Reporting

The reporting phase represents the final stage of the penetration testing process, aimed at compiling, summarizing, and presenting all testing results in a systematic and comprehensive manner. At this stage, findings from each PTES phase, including Pre-Engagement, Intelligence Gathering, Threat Modeling, Vulnerability Analysis, Exploitation, and Post-Exploitation, are consolidated into a report that provides a complete overview of the OJS application's security posture. The researcher prepared several final report documents, including a Penetration Testing Report containing all identified findings primarily intended for the client, a Final Report of the PTES Methodology, a Final Report of the OWASP WSTG Methodology, as well as separate documentation for each successfully exploited attack scenario. These reports present the risk level of each identified security gap, their impact on the confidentiality, integrity, and availability of the system based on the CIA framework, and prioritized remediation recommendations that can be used by system administrators as a structured reference to improve the application's security posture effectively and measurably.

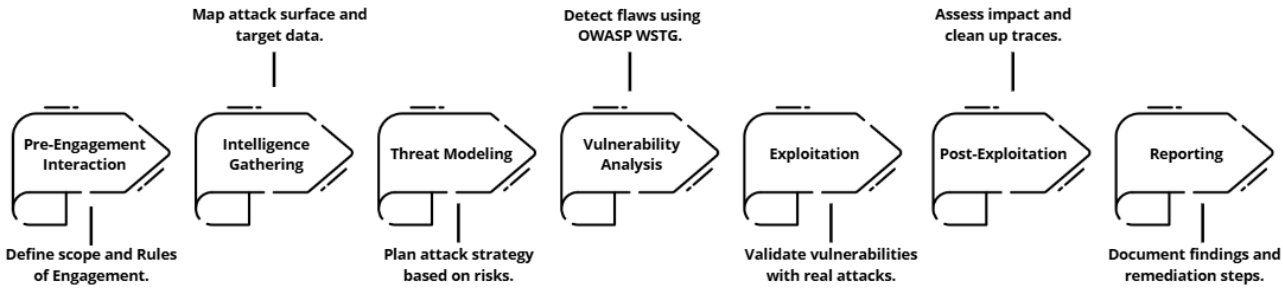


Image 3.1 – Flowchart

4. Result and Discussion

The security assessment of *xxx.com* was executed following the PTES (Penetration Testing Execution Standard) methodology and the OWASP Web Security Testing Guide (WSTG). This systematic approach resulted in a comprehensive evaluation of the platform's security posture through the successful completion of the pre-engagement, intelligence gathering, and threat modeling phases. In the exploitation phase, the researcher executed a total of eleven planned attack scenarios. Based on the statistical distribution of the findings, the assessment identified a total of seven distinct vulnerabilities categorized by their severity levels. As illustrated in the risk profile (Image 4.1), 29% of the identified risks are classified as Critical, 14% as High severity, and the remaining 57% as medium severity. This quantitative distribution indicates a critical vulnerability density that requires immediate remediation to prevent unauthorized system compromise. Overall, the initial phases successfully mapped the attack surface, providing a solid foundation for subsequent analysis.

In terms of technical analysis, the transition from SAST and DAST scanning to manual source-code review revealed significant security weaknesses. The assessment identified a lack of robust authentication safeguards, such as the continued use of the obsolete SHA-1 hashing algorithm, the absence of strict input validation mechanisms, and multiple information disclosure flaws. These weaknesses, including SQL Injection and Brute Force vulnerabilities, could enable unauthorized actors to bypass security controls or perform illicit database manipulations. Furthermore, the infrastructure assessment uncovered vulnerabilities related to resource management and server-side configurations. Such deficiencies, including the absence of rate-limiting mechanisms, expose the system to resource exhaustion attacks that directly threaten platform availability. The researcher also observed that several legacy components lack proper security headers and rely on outdated libraries, thereby increasing the likelihood of client-side attacks. Collectively, these technical findings indicate that the target environment contains multiple exploitable entry points.

The discussion of these results highlights a significant risk to organizational data integrity and regulatory compliance. The presence of critical vulnerabilities demonstrates that the current security posture is insufficient to adequately protect sensitive academic assets. Successful exploitation of these flaws could result in data leakage, erosion of user trust, and damage to the professional reputation of the academic platform. Moreover, unauthorized manipulation of academic content could cause irreversible harm to the journal's credibility within the scientific community. Therefore, this assessment underscores the urgent necessity for a prioritized remediation strategy. Key recommendations include upgrading authentication mechanisms to modern cryptographic standards, enforcing SSL/TLS encryption, and implementing comprehensive security headers. Strengthening these areas will significantly enhance the system's resilience against future threats and support long-term operational stability.

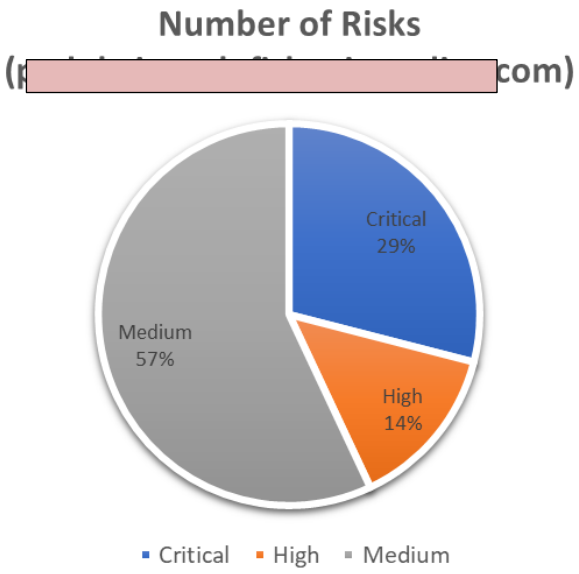


Image 4.1 – Graphic Results

5. Conclusion and Suggestion

5.1 Conclusion

Based on the entire series of security testing activities conducted on the *xxx.com* platform, it can be concluded that the system currently exhibits a vulnerable security posture. In this study, the researcher executed eleven (11) different attack scenarios, of which seven (7) vulnerabilities were successfully validated as exploitable attack vectors. The evaluation results indicate a high density of security weaknesses, with approximately 29% of the findings classified as Critical and 14% categorized as High severity. Several key factors were identified as significantly weakening the system's defensive capabilities, including the use of the obsolete SHA-1 hashing algorithm, the presence of empty API Key Secret values, and the deactivation of SSL encryption features. The successful exploitation of vulnerabilities such as SQL Injection and Brute Force attacks clearly demonstrates that sensitive assets, including user credentials and journal manuscript data, are at substantial risk of unauthorized access. Overall, although the initial testing phases effectively mapped the system's attack surface, the platform requires immediate remediation measures to mitigate these vulnerabilities and prevent potential data breaches that could undermine the credibility and professionalism of the academic journal system.

5.2 Suggestion

As a mitigation of the identified security findings, the researcher recommends upgrading the authentication mechanism by replacing the deprecated SHA-1 hashing algorithm with a modern and cryptographically secure standard such as SHA-256 or bcrypt. To address the exposure risk of API Key Secrets, all sensitive credentials must be stored securely using environment variables or dedicated secret management services and rotated periodically. Enforcing SSL/TLS across all application endpoints, particularly authentication and API communication channels, is essential to prevent credential interception during transmission. In addition, a configuration hardening process should be applied by disabling Directory Browsing on public directories and ensuring that security headers are properly defined to mitigate Clickjacking and XSS risks. Outdated third-party libraries, including Chart.js, must be updated to eliminate Prototype Pollution vulnerabilities. Lastly, the implementation of Rate Limiting mechanisms and the reactivation of reCAPTCHA are strongly recommended to reduce the risk of automated attacks targeting authentication and API access.

6. References

Alhogail, A., & Alkahtani, M. (2024). Automated extension-based penetration testing for web vulnerabilities. *Procedia Computer Science, 238*, 15–23. https://doi.org/10.1016/j.procs.2024.01.003

Anisah, S., & Aslamiyah, S. (2025). Website security analysis using penetration testing method. *Journal of Intelligent Decision Support System (IDSS), 8*(1), 1–8. https://idss.iocspublisher.org/index.php/jidss/article/view/284

Lachkov, P., Tawalbeh, L., & Bhatt, S. (2022). Vulnerability assessment for application security through penetration simulation and testing. *Journal of Web Engineering, 21*(7), 2187–2208. https://doi.org/10.13052/jwe1540-9589.2178

Kongara, D., & Krishnama, S. (2023). A process of penetration testing using various tools. *Mesopotamian Journal of Cybersecurity, 2023*, 93–103. https://www.researchgate.net/publication/373041127

Sari, D. P., & Pakaja, F. A. (2024). Carrying out website security analysis using the standard penetration testing method. *International Journal of Multidisciplinary Applied and Science Research, 1*(1), 22–28. https://www.researchgate.net/publication/393316877

Goel, J. N., & Mehtre, B. M. (2015). Vulnerability assessment & penetration testing as a cyber defence technology. *Procedia Computer Science, 57*, 710–715. https://doi.org/10.1016/j.procs.2015.07.458

Altulaihan, E. A., Alismail, A., & Frikha, M. (2023). A survey on web application penetration testing. *Electronics, 12*(5), 1229. https://doi.org/10.3390/electronics12051229

OWASP Foundation. (2023). OWASP Web Security Testing Guide (WSTG) v4.2. https://owasp.org/www-project-web-security-testing-guide/

OWASP Foundation. (2024). Penetration testing methodologies. In *OWASP Web Security Testing Guide (WSTG) v4.1*. https://owasp.org/www-project-web-security-testing-guide/v41/3-The_OWASP_Testing_Framework/1-Penetration_Testing_Methodologies

Penetration Testing Execution Standard (PTES). (n.d.). Penetration Testing Execution Standard. http://www.pentest-standard.org/index.php/Main_Page

Netsparker Security Team. (2020). The cross-site scripting (XSS) vulnerability: Definition and prevention. https://www.netsparker.com/blog/web-security/cross-site-scripting-xss

Bach-Nutman, M. (2020). Understanding the top 10 OWASP vulnerabilities. arXiv preprint arXiv:2012.09960. https://arxiv.org/abs/2012.09960

7. Appendix

7.1 Penetration Testing Report

https://polibatam.id/PenetrationTestingReport

7.2 Poster

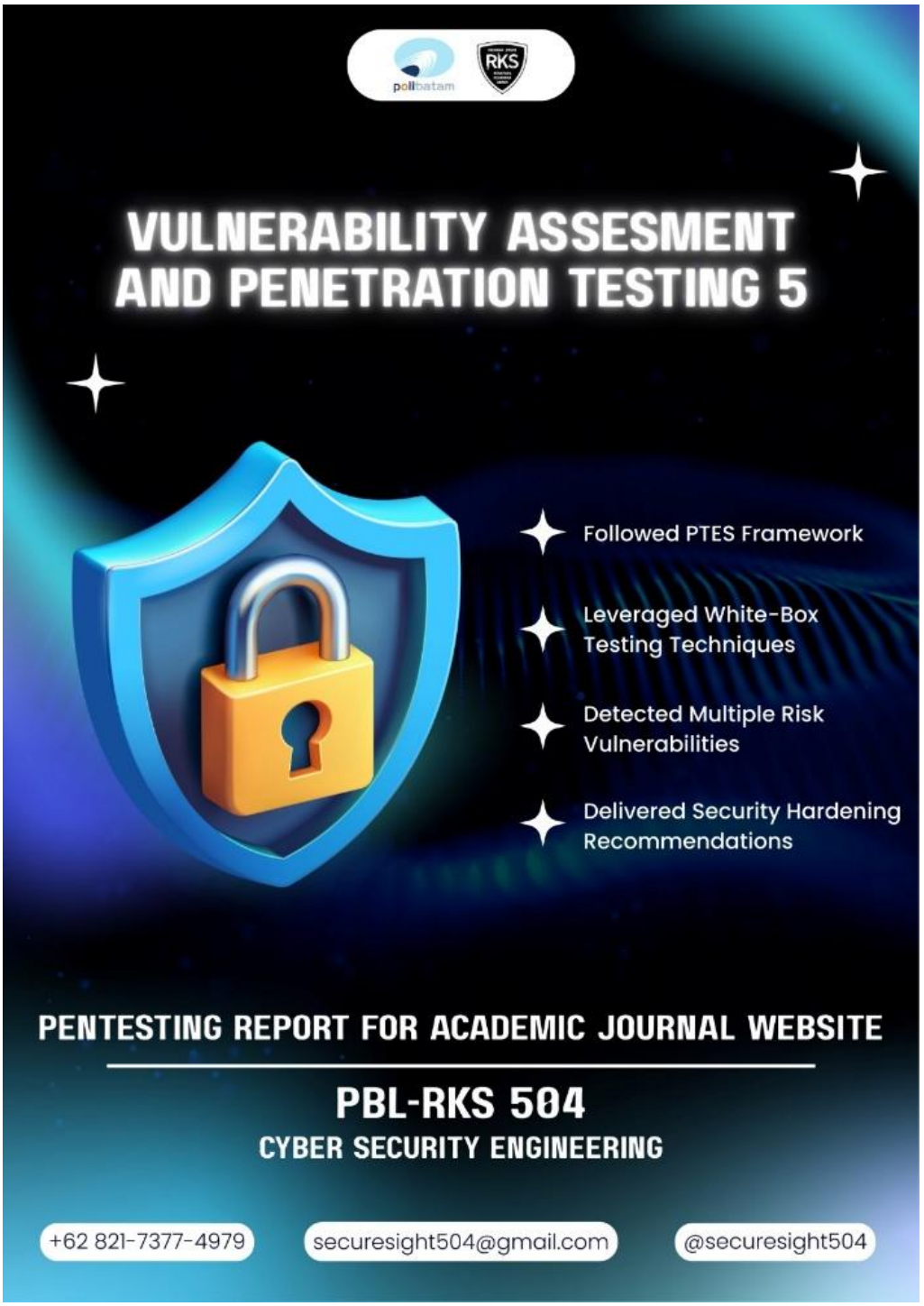


Image 7.1 Poster