Engineering and Technology Journal e-ISSN: 2456-3358

Volume 11 Issue 05 May-2026, Page No.-9680-9689

DOI: 10.47191/etj/v11i05.04, I.F. – 8.595

Security Evaluation of the TIF Dashboard Using the Penetration Testing Execution Standard (PTES) Methodology

Muhammad Aditya Rinaldi[1], Rahmad Abdillah[2*], Novriyanto[3], Pizaini[4]

[1,2,3,4]Department of Informatics Engineering, Faculty of Science and Technology, Universitas Islam Negeri Sultan Syarif Kasim Riau, Indonesia

ABSTRACT: The Dashboard TIF of UIN Suska Riau is an academic information system used to manage memorization submission data, murojaah history, and administrative processes of the Informatics Engineering Study Program. This system stores sensitive data such as student identification numbers (NIM), lecturer identification numbers (NIP), and students' memorization records, making its security highly critical. This study aims to evaluate the security level of the Dashboard TIF using the Penetration Testing Execution Standard (PTES) method combined with the OWASP Top 10 2025 framework. The assessment is carried out using a black-box approach with a combination of automated scanning tools and manual testing to discover, validate, and exploit relevant vulnerabilities. The results show that no critical vulnerabilities such as Remote Code Execution (RCE), SQL Injection, or Cross-Site Scripting (XSS) were found that could be directly exploited, but four categories of vulnerabilities requiring immediate remediation were identified, namely Security Misconfiguration (A02:2025), Software Supply Chain Failures (A03:2025), Insecure Design (A06:2025), and Authentication Failures (A07:2025). The most significant finding is an authentication weakness that allows brute-force attack against student accounts, which can be abused to access and modify academic data. The recommended improvements include strengthening security configurations, updating software components, and implementing authentication protection mechanisms to enhance the resilience of the Dashboard TIF against cyber attacks.

KEYWORDS: Penetration Testing, PTES, OWASP Top 10 2025, Web Application Security, Security Misconfiguration, Authentication Failures

1. INTRODUCTION

The rapid development of information technology in Indonesia education has led to increasing adoption of web-based systems to support academic activities. The Informatics Engineering Study Program at UIN Suska Riau exemplifies this trend through the Dashboard TIF, which manages critical data including student memorization submission and thesis administration. Dashboard TIF stores sensitive data such as Student ID (NIM) and Lecturer ID (NIP) used by both faculty and students, making its security critically important.

The growing use of web applications in education correlates with escalating cyber threats. The National Cyber and Crypto Agency (BSSN) reported 3.64 billion cyber attacks in Indonesia from January to July 2025, with the education sector identified as highly vulnerable [1]. The 2025 Verizon Data Breach Investigation Report (DBIR) indicates educational institutions globally average over 4,000 weekly security incidents [2]. Katadata 2025 confirms students and academics as the most frequent digital attack victims in Indonesia during Q1 2025 [3].

Previous studies have applied PTES methodology to Indonesian educational websites. Nur Fikri (2023) identified 11 vulnerabilities on SMA Negeri 1 Sokaraja website, including exploitable SQL Injection [4]. Yuwono (2026) discovered Broken Access Control and Insecure Direct Object Reference (IDOR) vulnerabilities [5]. Ditendra (2022) identified XSS and Brute Force flaws in academic information systems [6]. However, most studies focused primarily on vulnerability identification without comprehensive exploitation, incomplete PTES phase implementation, and reliance on previous OWASP Top 10 versions. Notably, no research specifically evaluated Dashboard TIF UIN Suska Riau Security.

This study aims to comprehensively assess Dashboard TIF UIN Suska Riau security using Penetration Testing Execution Standard (PTES) combined with OWASP Top 10 2025 through all seven phases: pre-engagement interactions, intelligence gathering, threat modeling, vulnerability analysis, exploitation, post exploitation, and reporting. PTES and OWASP Top 10 combination proves effective for structured vulnerability identification and categorization per industry standard [7]. Black-box testing identifies, analyses, and exploits system vulnerabilities. Findings are expected to provide realistic security assessment and actionable remediation recommendations for Dashboard TIF administrators.

2. LITERATURE REVIEW

2.1 Security Threats

The massive amount of information exchanged on the internet every second increases the risk of data being stolen or misused by unauthorized parties. According to W. Stallings, as cited in Raharjo [8], security threats to information systems can be grouped into four main categories: interruption, interception, modification, and fabrication. Interruption targets system availability and occurs when system devices or services are damaged, disabled, or made unavailable, causing data and information to become inaccessible to legitimate users.

Interception threatens confidentiality and occurs when an unauthorized party successfully eavesdrops on or gains access to information or to a computer where the information is stored. Modification threatens integrity and occurs when an unauthorized party intercepts data in transit and then alters or manipulates its contents according to the attacker's objectives. Fabrication also threatens integrity and occurs when an attacker forges or injects false information into a system so that the recipient believes the information comes from a legitimate source.

2.2 Vulnerability

A vulnerability represents a weakness in an asset or group of assets that can be exploited by one or more threats. According to the IETF (Internet Engineering Task Force) definition in RFC 2828, a vulnerability is a flaw or weakness in a system whether in design, implementation, operation, or management that can be exploited to attack an information system [9].

2.3 Common Vulnerabilities and Exposures (CVE)

Common Vulnerabilities and Exposures (CVE) is a standardized identification system for publicly known cybersecurity vulnerabilities. The CVE program is maintained by The MITRE Corporation and sponsored by the U.S. Cybersecurity and Infrastructure Security Agency (CISA) and the Department of Homeland Security, and was officially launched in September 1999 [10]. Each CVE entry consists of a unique identifier in the format CVE-[Year]-[Number], a brief description of the vulnerability, and references to technical details or available patches. The CVE system provides a public "dictionary" of vulnerability names, enabling security researchers, vendors, and practitioners to refer to the same vulnerability consistently and without ambiguity [11].

The severity of a CVE is assessed using the Common Vulnerability Scoring System (CVSS), a numerical score ranging from 0.0 to 10.0 developed by FIRST (Forum of Incident Response and Security Teams). Higher CVSS scores indicate more dangerous and easily exploitable vulnerabilities. CVEs are listed in both the MITRE system and the National Vulnerability Database (NVD) maintained by the National Institute of Standards and Technology (NIST), making them publicly accessible to anyone for security research and mitigation purposes [12].

2.4 Penetration Testing Execution Standard (PTES)

The Penetration Testing Execution Standard (PTES) is an international standard framework used as a systematic and professional guide for conducting penetration testing (pentest). PTES is particularly effective for web applications as it can be easily adopted by users with limited penetration testing experience [13]. PTES divides the entire testing process into seven main sequential and interconnected phases, as shown in figure 1 [14]. These seven phases ensure testing is conducted in a structured, ethical, and accountable manner. The following provides an explanation of each phase:

Figure 1. PTES Phase

1. Pre-engagement Interaction Phase, this planning phase involves initial communication with system owners to define testing scope, objectives, Rules of Engagement, time constraints, and legal considerations. This phase ensures testing proceeds safely, legally, and according to expectations.
2. Intelligence Gathering Phase, in this phase, testers collect as much target information as possible, including domains, subdomains, IP addresses, technologies used, organizational structure, employee email addresses, and other public information. Information gathering is divided into passive reconnaissance (no direct target contact) and active reconnaissance (port scanning, service enumeration). This phase is crucial as more comprehensive intelligence increases the likelihood of identifying security flaws in subsequent phases.
3. Threat Modeling Phase, this phase aims to identify and prioritize the most likely threats to the system. Testers analyze collected intelligence to create threat models considering potential attackers, their objectives, and attack methods. Threat modeling focuses testing efforts on highest-risk areas.
4. Vulnerability Analysis Phase, testers conduct in-depth scanning and analysis to identify security flaws in the target system. Activities include port

scanning, service enumeration, software version identification, and manual testing of web applications, APIs, and infrastructure. Each identified vulnerability is then risk-assessed. This phase bridges information gathering and attack execution.

5. Exploitation Phase, this core penetration testing phase attempts to exploit identified vulnerabilities to gain unauthorized system access. Exploitation may involve SQL Injection, XSS, brute force, buffer overflow, or other techniques. The primary goal is to prove vulnerabilities can be successfully exploited with real impact. All activities are conducted cautiously to avoid damaging production systems.
6. Post-Exploitation Phase, after gaining system access, testers proceed to post-exploitation analysis. Activities include pivoting (movement to other systems on the same network), privilege escalation (elevating access rights), sensitive data collection, and persistence (maintaining access). This phase assesses how extensively attackers can navigate and damage systems post-compromise, evaluating overall vulnerability impact.
7. Reporting Phase, the final phase involves compiling comprehensive test results. Reports must include executive summaries (for management), detailed vulnerability findings, risk levels, technical evidence, and clear, actionable remediation recommendations. PTES reports must be objective, accessible, and serve as references for stakeholders to improve systems. This phase represents the most critical outcome of the entire testing process.

2.5 OWASP Top 10 2025

The Open Web Application Security Project (OWASP) is an open community dedicated to improving web application security through individual, process, and technology approaches [15]. OWASP Top 10 2025 represents the top 10 web application security risks based on real incidents from thousands of applications worldwide. Implementing OWASP Top 10 as a standard reference for penetration testing proves effective in identifying and evaluating significant security flaws in systems, while providing clear direction for enhancing security in academic information system websites [16]. The OWASP Top 10 2025 categories are shown in Figure 2.

Figure 2. OWASP Top 10 2025

OWASP Top 10 2025 comprises the 10 most critical web application security risks. The detailed explanations are as follows:

1. A01:2025 – Broken Access Control, this category ranks first as the most serious web application security risk. Broken Access Control occurs when the system fails to enforce authorization rules, allowing users to access data or features they should not be permitted to access. For example, regular users can view other users' data or perform administrator actions. This attack is common because developers often only check access rights on the client side (frontend), whereas actual verification must occur on the server side.
2. A02:2025 – Security Misconfiguration, this category rose to second place in 2025. Security Misconfiguration occurs when systems, applications, or cloud services are configured insecurely, such as using weak default settings, leaving debugging features active in production, or failing to restrict access to sensitive files and directories. The increasing complexity of modern applications makes these configuration errors more frequent.
3. A03:2025 – Software Supply Chain Failures, a new category entering third place. This expands on "Vulnerable and Outdated Component" to include vulnerabilities or attacks on third-party dependencies, CI/CD Pipelines, build processes, or software distribution. A single compromised component can affect entire applications or even thousands of other applications using the same library.
4. A04:2025 – Cryptographic Failures, dropped to fourth place. Occurs when encryption hashing, or key management mechanisms are implemented incorrectly, allowing unauthorized parties to read sensitive data. Examples include using outdated encryption algorithms or storing sensitive data in plain text.

5. A05:2025 – Injection, attacks where adversaries inject malicious commands or queries into application inputs (such as SQL Injection, command Injection, or XSS). SQL Injection remains a serious threat as many applications fail to properly validate and sanitize inputs.
6. A06:2025 – Insecure Design, dropped to sixth place. Insecure Design refers to weaknesses present from the application design phase, not implementation errors. Examples include lack of threat modeling, use of insecure design patterns, or failure to consider attack scenarios during initial development.
7. A07:2025 – Authentication Failures, remains in seventh place with a slight name change from "Identification and Authentication Failures." This encompasses issues in login processes, session management, and user identity verification.
8. A08:2025 – Software and Data Integrity Failures, remains in eighth place. Focused on failures to maintain code authenticity and integrity, software update, and critical data. Examples include not verifying digital signatures on updates or downloaded libraries, allowing attackers to inject malicious code.
9. A09:2025 – Security Logging and Alerting Failures, remains in ninth place. Although logging is implemented, effective alerting mechanisms are often absent. Consequently, ongoing attacks go undetected in real-time, making investigation and incident response difficult.
10. A10:2025 – Mishandling of Exceptional Conditions, a new category entering tenth place in 2025. Addresses failures in handling errors, abnormal conditions, or exceptional situations in applications. Examples include displaying overly detailed error messages (leaking system information), systems that "fail open", or broken application logic when exceptions occur.

3. METHODOLOGY

This study applies the Penetration Testing Execution Standard (PTES) combined with the OWASP Top 10 2025 framework. The assessment is conducted using a black-box approach, meaning the tester has no prior knowledge of the source code structure or initial configuration of the target system. The research object is the Dashboard TIF of UIN Suska Riau. The research methodology follows the seven main PTES phases:

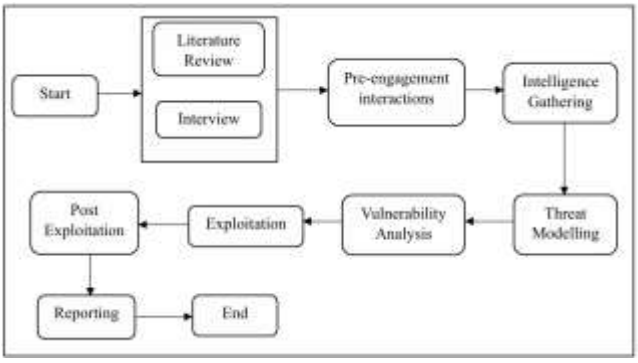

Figure 3. Research Methodology

3.1 Data collection

The data collection phase aims to gather information required for the research process. In this study, data collection was conducted by obtaining relevant information from books, journals, theses, and articles supporting the research. Additional data collection involved interviews with the Dashboard TIF Project Manager (PM) through a question and answer process to acquire high quality information.

Dashboard TIF is an integrated academic administration service for the Informatics Engineering (TIF) Study Program, operational since January 2025. It supports memorization submission services for Juz 30 (from surah Ad-Dhuha to Surah An-Nas). Future expansion will include work practice and thesis administration services, covering work practice registration, thesis title proposal, and seminar registration. Dashboard TIF stores sensitive data such as lecturer ID (NIP) and student ID (NIM) as it is used by both faculty and students.

Figure 4. Dashboard TIF Interface

Dashboard TIF was developed using Node.js for the backend, PostgreSQL for the database, and Keycloak for authentication. The system also implements a firewall, server-side logging to record and monitor all activities, and conducts routine updates for both the application and server.

3.2 Pre-engagement Interaction

This phase involves initial communication between the tester and Dashboard TIF administration to define testing scope, objectives, limitations, and rules of engagement. During this phase, an interview was conducted with the Dashboard TIF Project Manager (PM) to obtain preliminary system information, including technologies used, stored data, and available features.

3.3 Intelligence Gathering

This phase aims to collect as much information as possible about the target before testing begins. Information gathered includes IP addresses, technologies used, SSL configuration, domain details. Information collection was performed using tools such as Reverse IP and Wappalyzer.

3.4 Threat Modeling

This phase aims to identify and model potential threats to Dashboard TIF based on information collected in previous phases. Threat Modeling was conducted using the OWASP Top 10 2025 framework as the primary reference for web application threat categories.

3.5 Vulnerability Analysis

This phase aims to identify and evaluate potential vulnerabilities in Dashboard TIF based on the modeled threat categories. Vulnerability identification was performed using three main tools: Nuclei, OWASP ZAP, Acunetix. Scan results from all three tools were validated to ensure identified vulnerabilities were legitimate before proceeding to the exploitation phase.

3.6 Exploitation

This phase aims to prove previously identified vulnerabilities through controlled exploitation attempts. Applied exploitation techniques include manual access control testing, SQL Injection using sqlmap, Cross-site Scripting (XSS) testing using Burp Suite, and brute force attacks using Python scripts.

3.7 Post Exploitation

This phase aims to measure the real impact resulting from successfully exploited vulnerabilities. Testing simulates actions that attackers might perform after gaining system access.

3.8 Reporting

This phase aims to systematically compile testing results. The report includes all vulnerabilities, risk levels, exploitation evidence, and remediation recommendations mapped to all OWASP Top 10 2025 categories. The report will be submitted to Dashboard TIF administrators as the basis for system improvements.

4. RESULT AND DISCUSSION

Based on testing conducted using the PTES methodology combined with the OWASP Top 10 2025 framework, the following presents the results and discussion from each testing phase.

4.1 Pre-engagement Interaction

The Pre-engagement Interaction phase involved obtaining authorization from Dashboard TIF administrators, specifically the Informatics Engineering Study Program, to conduct the security evaluation. Discussions covered testing timelines, testing boundaries, and reporting procedures for identified vulnerabilities.

4.2 Intelligence Gathering

This phase focused on searching and collecting critical information about Dashboard TIF to serve as a reference for the security evaluation. The first step involved identifying the Dashboard TIF IP address and checking for subdomains using reverse IP lookup website. Results revealed the Dashboard TIF IP address as 103.xxx.xx.xx, with no subdomains detected as shown in Figure 5.

Figure 5. Dashboard TIF IP Address Lookup

After obtaining the IP address, technology stack identification was performed using the Wappalyzer browser extension. Results are shown in Figure 6.

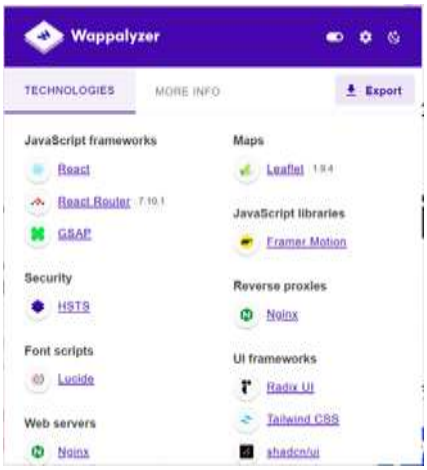

Figure 6. Dashboard TIF Technology Identification

4.3 Threat Modeling

Based on its characteristics of storing NIM, NIP, and memorization submission records, threat modeling uses the OWASP Top 10 2025 framework to evaluate the security of Dashboard TIF, because these ten threat categories occur most frequently and the OWASP Top 10 is commonly used to identify vulnerabilities in web-based applications [17].

4.4 Vulnerability Analysis

The Vulnerability Analysis phase aimed to identify vulnerabilities in Dashboard TIF for use in subsequent phases. This phase employed Nuclei, ZAP, and Acunetix tools to detect existing vulnerabilities. Nuclei scanning revealed Dashboard TIF uses a Web Application Firewall (WAF), implements SSL (Secure Socket Layer), and identified security misconfiguration header vulnerabilities.

Nuclei also detected a publicly accessible admin login page that should not be exposed, as shown in Figure 7 [18].

Figure 7. Nuclei Scanning Results

ZAP Active Scan identified vulnerabilities comprising 1 High risk, 4 Medium risk, and 4 Low risk vulnerabilities, mapping to 1 vulnerability in A01:2025 – Broken Access Control category and 8 vulnerabilities in A02:2025 – Security Misconfiguration category. ZAP scan results are presented in Table 1. Acunetix scanning produced findings consistent with Nuclei and ZAP, confirming header misconfiguration vulnerabilities.

Table 1. ZAP Scan Results

Vulnerability	Risk Level	Category
Off-site Redirect	High	A01
CSP Header Not Set	Medium	A02
Missing Anti Clickjacking Header	Medium	A02
Absence of Anti-CSRF Tokens	Medium	A02
Cross-Domain Misconfiguration	Medium	A02
Server leaks information	Low	A02
Cookie No HttpOnly Flag	Low	A02
Strict transport security header	Low	A02
X-content type header missing	Low	A02

4.5 Exploitation

Based on the vulnerabilities identified in the previous phase, the exploitation phase was carried out to verify whether these vulnerabilities were truly exploitable. The first vulnerability tested was the Off-site Redirect with a High risk level. Testing was performed on the parameter post-logout-redirect=https://dashboard.xxx.xxx-xxx.ac.id by replacing it with a simple payload, https://www.example.com/. When the payload was submitted and executed, the logout page refused to redirect to Google and instead displayed a message as shown in Figure 8. This indicates that the Off-site Redirect alert reported by ZAP was a false positive, where ZAP treated post-logout-redirect=https://dashboard.xxx.xxx-xxx.ac.id as a potentially dangerous value and raised an alert.

Figure 8. Off-site Redirect Test

Next, a Broken Access Control test was performed. This attack targets systems that fail to properly enforce authorization rules, allowing unauthorized users to access pages and modify data outside their privileges. On Dashboard TIF, this was tested by logging in using a student account and attempting to access a lecturer page, namely "dosen/xxxx/xxxx", which is used to record memorization submissions for the lecturer's academic advisees.

The results showed that when a student attempted to access the lecturer page, the system denied access and displayed an "access denied" message because the student did not have permission to view that page. This demonstrates that Dashboard TIF has implemented access control correctly, as users can only access authorized pages, as shown in Figure 9.

Figure 9. Unauthorized Access Attempt

The next stage involved testing injection attacks, which occur when attackers inject malicious code into application inputs that are not properly sanitized. Two types of injection attacks were tested: SQL Injection and Cross-Site Scripting (XSS).

The first injection test was SQL Injection, aiming to access the Dashboard TIF database using sqlmap. The test results showed that SQL Injection attempts failed, partly due to Dashboard TIF being protected by a Web Application Firewall (WAF) that blocked sqlmap's attack patterns. The results are shown in Figure 10.

Figure 10. SQL Injection Test Using SQLMap

The second injection test was Cross-Site Scripting (XSS). This attack typically involves injecting malicious code into a legitimate system which, if input sanitization is insufficient, may be executed to steal cookies, login sessions, or other sensitive data. Since Dashboard TIF is designed with minimal user input fields apart from the login page, testing was conducted against the login API using Burp Suite.

The test attempted to inject the payload "<script>alert(1)</script>" into the state parameter. If successfully executed, a "1" popup would appear on Dashboard TIF. However, when tested, the login page loaded normally and the script was not executed. This indicates that XSS could not be performed and Dashboard TIF is not vulnerable to XSS in this context, demonstrating proper input sanitization.

The next test was a brute force attack on both user and admin Keycloak login pages using Burp Suite and a custom Python script with simple credential payloads. The first brute force attempt used Burp Suite Intruder on student/lecturer and admin login pages to test whether default credentials were still in use and whether any rate limiting mechanisms were present on the login endpoint.

When executed, the brute force process indicated that repeated login attempts were allowed, suggesting the absence of rate limiting to prevent brute force attacks. However, Burp Suite Intruder failed to obtain valid student/lecturer or admin credentials because it only modified the username and password parameters, while dynamic session parameters in Keycloak such as session_code, execution, tab_id, and client_data were not refreshed on each attempt.

According to the official Keycloak documentation, the Login Action Timeout controls the maximum allowed time for each page in the authentication flow [19], and community discussions note that the default value is 1 minute [20]. In practice, session parameters on Dashboard TIF expired within 15–30 seconds. Keycloak requires each POST request in the authentication process to use fresh session parameter values, and because these parameters are one-time use, all repeated requests with stale parameters were considered invalid, causing Keycloak to consistently return HTTP 302 responses for each attempt.

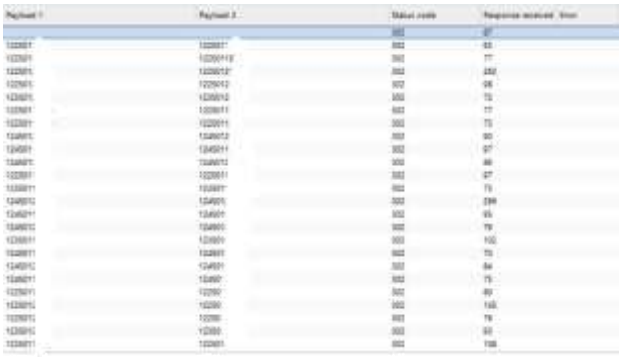

Figure 11. Brute Force Attack Using Burp Suite

Based on insights gained from the brute force attempt using Burp Suite Intruder, a custom Python script was developed to repeatedly obtain new session values during brute force attempts, refreshing the session for each new username and password combination. Using this approach, brute force attacks on both student/lecturer and Keycloak admin login pages were successfully executed, confirming the absence of rate limiting, CAPTCHA, or arithmetic challenges to block brute force attacks. Although the tests did not manage to obtain Keycloak admin or lecturer credentials, they successfully discovered multiple valid student credentials, indicating that many students still used default credentials on Dashboard TIF.

This condition occurs because Dashboard TIF does not yet provide a password change feature and does not enforce or encourage users to update their default passwords after account activation. Successful brute force results are shown in Figure 12.

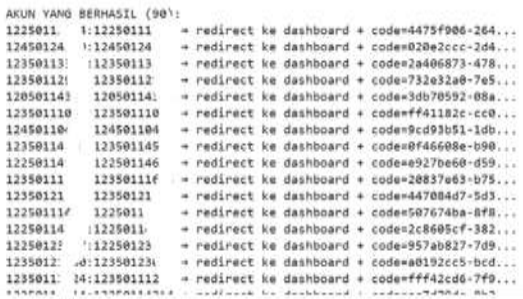

Figure 12. Brute Force Results

Another identified issue was HTTP security header misconfiguration, which could not be directly exploited during testing but may facilitate future attacks. Missing or improperly configured headers can increase exposure to attacks such as XSS and cookie theft. Therefore, these misconfigurations still require remediation to reduce the attack surface and prevent potential future exploits.

Figure 13. Missing Header Configuration

4.6 Post Exploitation

From the previous brute force attack phase, which successfully obtained student accounts, this phase utilized those compromised credentials to perform unauthorized login attempts. The results, shown in Figure 14, demonstrate that login was successful, granting the attacker access to sensitive data such as memorization submission records. These records could potentially be modified by the attacker, enabling the completion or falsification of administrative documentation managed by the Informatics Engineering Study Program. This post-exploitation scenario underscores the significant risk of unauthorized data manipulation following initial credential compromise.

Figure 14. Login Attempt

4.7 Reporting

Based on the previous PTES phases, the next step involves preparing this report according to the threat modeling framework used, namely OWASP Top 10 2025.

A. A01:2025 – Broken Access Control: No vulnerabilities were found after manual testing using a student account to access lecturer-only pages. The system correctly rejected access and displayed the message "Sorry, you do not have access to this page," indicating proper role and access control configuration, as shown in Figure 9. Testing of the Off-site Redirect flagged by ZAP was also confirmed as a false positive after manual verification, with results shown in Figure 8.

B. A02:2025 – Security Misconfiguration: Vulnerabilities were identified, including missing security headers and a publicly accessible Keycloak admin/master login page, indicating configuration weaknesses. Recommended remediation includes implementing comprehensive security headers and restricting public access to the admin login page, allowing only trusted networks.

C. A03:2025 – Software Supply Chain Failure: outdated libraries were identified, specifically React Router version 7.10.1, which carries CVE-2025-59057 (XSS via the meta function, CVSS 6.1 Medium) and CVE-2025-43865 (pre-render data spoofing, CVSS Medium), as well as Leaflet version 1.9.4, which carries CVE-2025-69993 (XSS via the bindPopup method, CVSS 6.1 Medium). Testing was conducted on Dashboard TIF and the CVEs could not be successfully exploited. Nevertheless, updating these libraries to their latest versions is strongly recommended to prevent potential exploitation as attack techniques continue to evolve.

D. A04:2025 – Cryptographic Failures: This category did not reveal any vulnerabilities, as Dashboard TIF already uses HTTPS with a valid SSL/TLS certificate and the cryptographic algorithms in use are still adequate. However, a recent check shows that the SSL/TLS certificate for Dashboard TIF is approaching its expiration date.

E. A05:2025 – Injection: No vulnerabilities were found after testing, as shown in Figure 10. Injection attempts failed, confirming effective input sanitization on Dashboard TIF.

F. A06:2025 – Insecure Design: A vulnerability exists due to the absence of rate limiting on login pages, enabling brute force attacks against user and admin/master Keycloak endpoints. Recommendations include enabling Keycloak's built-in brute force protection [21] or implementing CAPTCHA and arithmetic challenges.

G. A07:2025 – Authentication Failures: Vulnerabilities were confirmed through successful brute force attacks due to no rate limiting, revealing multiple students using default credentials, as shown in Figure 12. This stems from the lack of password change features and guidance post-account activation, allowing attackers to log in and access submission records. Recommendations include adding password reset functionality and notifications urging users to change passwords upon activation.

H. A08:2025 – Software or Data Integrity Failures: No vulnerabilities were identified in this category, as Dashboard TIF does not permit the execution of suspicious code.

I. A09:2025 – Security Logging and Alerting Failures: No vulnerabilities were found, based on interviews confirming that logs are directly managed by the Informatics Engineering Study Program at UIN Suska Riau.

J. A10:2025 – Mishandling of Exceptional Conditions: No vulnerabilities such as information-leaking error messages were detected. Dashboard TIF demonstrates effective input sanitization following testing.

V. CONCLUSIONS

This research aimed to evaluate the security of Dashboard TIF at UIN Suska Riau using the Penetration Testing Execution Standard (PTES) methodology combined with the OWASP Top 10 2025 framework across seven black-box testing phases. Dashboard TIF is a web application designed to support administrative services for the Informatics

Engineering Study Program, managing sensitive data such as student and lecturer information alongside memorization submission records essential for administrative processes. Thus, its security is critically important.

Testing results indicate that Dashboard TIF exhibits no critical vulnerabilities such as Remote Code Execution (RCE), SQL Injection, or Cross-Site Scripting (XSS) that could be directly exploited at the application level.

However, four vulnerability categories require immediate remediation: A02:2025 – Security Misconfiguration (missing security headers and improper cookie configuration), A03:2025 – Software Supply Chain Failure (outdated libraries with registered CVEs), A06:2025 – Insecure Design (lack of rate limiting on login pages), and A07:2025 – Authentication Failures (numerous accounts using default credentials successfully obtained via brute force). The A07:2025 findings represent the highest risk, as they enable attackers to access the system and misuse student academic data.

Based on these findings, Dashboard TIF administrators are recommended to promptly implement comprehensive security headers, update libraries to the latest versions, enable Keycloak's brute force protection or add CAPTCHA mechanisms, and introduce password change features with notifications urging users to update credentials upon account activation.

Additionally, it is recommended to implement an email-based security notification feature, specifically an automatic alert sent to the user's registered Gmail or email address whenever a login from a new device or unrecognized location is detected. This feature is a standard security practice widely adopted by digital platforms to enable early detection of unauthorized access and allow users to take immediate action, such as securing their account or reporting the incident to the system administrator.

Due to limited access to Dashboard TIF's source code, future research should conduct white-box testing with full source code and server configuration access to uncover vulnerabilities missed in black-box testing. Regular security evaluations are also advised to ensure ongoing protection against evolving cyber threats.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the Project Manager and the development team of Dashboard TIF, Informatics Engineering Study Program, Universitas Islam Negeri Sultan Syarif Kasim Riau, for granting permission and providing full support throughout the security evaluation process. The authors also extend their appreciation to the supervisors and all parties who contributed to the completion of this research.

REFERENCES

1. Tempo, "BSSN Catat 3,64 Miliar Serangan Siber di Indonesia Setengah Tahun Ini," tempo.co. Accessed: Oct. 31, 2025. [Online]. Available: https://www.tempo.co/digital/bssn-catat-3-64-miliar-serangan-siber-di-indonesia-setengah-tahun-ini-2056396
2. Keepnets, "2025 Verizon Data Breach Investigations Report," keepnetlabs.com. Accessed: Oct. 31, 2025. [Online]. Available: https://keepnetlabs.com/blog/2025-verizon-data-breach-investigations-report
3. A. Ridwan, "Jumlah Serangan Digital di Indonesia Kuartal I 2025, Korbannya Mayoritas Pelajar," katadata.co.id. Accessed: Oct. 31, 2025. [Online]. Available: https://databoks.katadata.co.id/teknologi-telekomunikasi/statistik/6847e2375639e/jumlah-serangan-digital-di-indonesia-kuartal-i-2025-korbannya-mayoritas-pelajar
4. M. Nur Fikri, B. Parga Zen, R. Adhitama, and E. Ahmad Firdaus, "Analisis Keamanan Sistem Informasi Website SMA Negeri 1 Sokaraja Menggunakan Metode Penetration Testing Execution Standard (PTES)," *J. Inform.*, vol. 2, no. 2, pp. 19–27, 2023, doi: 10.57094/ji.v2i2.1046.
5. Y. Y. Yuwono, D. Ferdiansyah, and M. F. Muttaqin, "Analisis Keamanan Website Universitas Pasundan Menggunakan Metode Penetration Testing Berbasis Kerangka Kerja PTES," vol. 5, no. 1, 2026.
6. Eghi Ditendra, "Evaluasi Keamanan Sistem Informasi Akademik Rokania Menggunakan Metode Penetration Testing Execution Standards(PTES)," UIN Suska Riau, Pekanbaru, 2022.
7. M. Y. Firnanda, H. E. Wahanani, and A. Junaidi, "ERP Website Security Testing Using PTES Method and OWASP Top 10 Approach," vol. 8, no. 1, p. 12, 2025, doi: 10.32877/bt.v8i1.2564.
8. B. Raharjo, *Keamanan Sistem Informasi Berbasis Internet*. Jakarta: PT Insan Infonesia - Bandung & PT INDOCISC, 2002.
9. M. Alhaqi, "Evaluasi Keamanan Website Terhadap Serangan Hacker Dengan Metode Penetration Test Menggunakan OWASP ZAP (Studi Kasus: SiPA Fakultas Teknik Universitas Islam Riau)," Universitas Islam Riau, 2024.
10. MITRE, "Common Vulnerabilities and Exposures (CVE)," cve.mitre.org. Accessed: Apr. 24, 2026. [Online]. Available: https://cve.mitre.org
11. P. Przymus, M. Fejzer, J. Narebski, and K. Stencel, "The Secret Life of CVEs," 2025, doi: https://doi.org/10.48550/arXiv.2504.03863.
12. NVD/NIST, "CVEs and the NVD Process," nvd.nist.gov. Accessed: Apr. 24, 2026. [Online]. Available: https://nvd.nist.gov/general/cve-process
13. T. R. Syarif and D. A. Jatmiko, "Analisis Perbandingan Metode Web Security PTES , ISSAF

dan OWASP di Dinas Komunikasi dan INFORMASI Kota Bandung Universitas Komputer Indonesia," p. 8, 2019, [Online]. Available: http://elibrary.unikom.ac.id/id/eprint/880

14. Ptes, "The Penetration Testing Execution Standard Documentation," 2022.
15. OWASP, "Owasp Top 10:2025." Accessed: Dec. 01, 2025. [Online]. Available: https://owasp.org/Top10/2025/
16. F. Tinambunan, A. Junaidi, and A. Mustika Rizki, "Pengujian Sistem Informasi Akademik Universitas X Melalui Pendekatan Penetration Testing Berdasarkan Owasp Top 10," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 1, pp. 1062–1069, 2024, doi: 10.36040/jati.v8i1.8920.
17. S. D. Raihan, S. Prabowo, and D. Oktaria, "Evaluasi Risiko Celah Keamanan Pada Aplikasi Web Dengan Penilaian Kerentanan dan Pengujian Penetrasi Menggunakan Metode OWASP dan NIST SP 800-30 Revisi 1," vol. 2, no. 2, pp. 27–34, 2024.
18. Keycloak, "Admin Endpoint and Admin Console," Keycloak Documentation. Accessed: Apr. 01, 2026. [Online]. Available: https://www.keycloak.org/docs/latest/server_admin/#admin-endpoints-and-admin-console
19. Keycloak, "Server Administration Guide: Session and Token Timeouts," Keycloak Documentation. Accessed: Apr. 04, 2026. [Online]. Available: https://www.keycloak.org/docs/latest/server_admin/#_timeouts
20. Keycloak User Group, "How to set authorization code timeout?," Google Groups. Accessed: Apr. 06, 2026. [Online]. Available: https://groups.google.com/g/keycloak-user/c/fGrUo0vqTPU?pli=1
21. Keycloak, "Brute force attacks," Keycloak Documentation. Accessed: Apr. 01, 2026. [Online]. Available: https://www.keycloak.org/docs/latest/server_admin/#password-guess-brute-force-attacks