International Journal of Applied Information Systems and Cybersecurity
Vol. 1 No. 1 (2026) pp. 1-18
Online ISSN: XXXX-XXXX

Original Research

Evaluating Web Security Resilience Using ISSAF: Evidence from a Public Village Information System

Putra Prasetya[1], Harjono[1]✉, Mukhlis Prasetyo Aji[1], Ermadi Satriya Wijaya[1]
[1] Universitas Muhammadiyah Purwokerto, Jawa Tengah, Indonesia
✉ harjono@ump.ac.id
10.65840/ijaisc.vxix.xx

Article Info
Submitted:
10/11/2025
Revised:
11/10/2025
Accepted:
12/20/2025
Available Online:
01/15/2026

Abstract. *The growing adoption of web-based public services has increased the cybersecurity risks faced by government information systems, particularly in small-scale environments with limited security resources. This study assesses the security of a village information system web application using the Information Systems Security Assessment Framework (ISSAF). The assessment was conducted through four stages: information gathering, network mapping, vulnerability assessment, and penetration testing. The findings indicate that the application has a moderate security posture. Information gathering revealed limited public exposure, although the lack of DNSSEC and accessible metadata presents potential security risks. Network mapping showed a small attack surface with only essential ports exposed. Vulnerability assessment identified several medium- and low-severity issues, mainly related to security misconfigurations such as missing HTTP security headers and weak cookie settings, while no critical vulnerabilities were detected. Penetration testing involving SQL injection, cross-site scripting (XSS), brute-force authentication, and denial-of-service simulations did not achieve successful exploitation, demonstrating the effectiveness of existing input validation and access control mechanisms. Overall, the application is resistant to common cyberattacks but remains vulnerable to multi-stage attacks caused by configuration weaknesses. These findings demonstrate the usefulness of ISSAF for evaluating web application security and provide practical recommendations for strengthening cybersecurity in resource-constrained e-government environments.*

Keywords: *Web application security; ISSAF; penetration testing; vulnerability assessment; cybersecurity; e-government; network security; ethical hacking*

1. Introduction

The increasing reliance on web-based systems across both public and private sectors has significantly expanded the digital attack surface. As organizations move essential services online, cybersecurity threats have evolved in parallel—becoming more frequent, more automated, and often more sophisticated. Web applications, in particular, remain a primary target due to their direct exposure to users and their frequent dependence on complex, and sometimes poorly maintained, configurations. Common weaknesses such as misconfigurations, outdated libraries, and improper input validation continue to be exploited in real-world attacks, often with severe consequences for data confidentiality and system integrity (B et al., 2024).

This risk becomes more pronounced in the context of government-operated systems. Public service websites are designed to be widely accessible, which inherently increases their exposure to potential threats. At the village level, information systems play a crucial role in delivering administrative services and disseminating public information. These platforms frequently handle sensitive administrative data, thereby increasing their attractiveness as targets for cyber attacks. Despite their operational importance, many small-scale government web applications function under limited technical and security resources and lack continuous security monitoring. As a result, vulnerabilities may persist undetected, not because they are complex, but because they are overlooked (Darojat et al., 2022).

To address such challenges, various security assessment frameworks have been proposed and widely adopted. Among them, the Information Systems Security Assessment Framework (ISSAF) and the Open Web Application Security Project (OWASP) provide structured approaches for evaluating web application security. These frameworks guide the assessment process through systematic stages, from reconnaissance to exploitation. Prior studies applying these methodologies have consistently identified vulnerabilities such as SQL injection, cross-site scripting (XSS), and configuration-related weaknesses across different environments (Wen & Katt, 2023). Their findings reinforce the importance of conducting regular and methodical security assessments.

However, a closer look at the literature reveals an imbalance. Most existing research focuses on large-scale environments—universities, enterprise systems, or cloud infrastructures—where resources and security practices are relatively mature. In contrast, small-scale public service systems, such as village information platforms, remain underexplored. These systems often operate under different constraints, where security is not always prioritized or systematically evaluated. This creates a gap in understanding how well such systems can withstand real-world attack scenarios.

In response to this gap, this study applies the ISSAF framework to evaluate the security posture of a village information system. The assessment is designed to move beyond simple vulnerability identification by examining the practical exploitability of detected weaknesses. By integrating multiple testing phases—ranging from information gathering to controlled attack simulation—this study aims to provide a comprehensive and realistic evaluation of system resilience. The findings are expected to contribute both empirically and practically, offering insights into how security can be improved in small-scale government web environments. The following sections present the research methodology, followed by the results and discussion of the security assessment, and conclude with key findings and implications.

2. Method

2.1. Research Design

To address the research objective outlined in the previous section, a structured security assessment methodology based on ISSAF was employed. This study adopts an experimental security evaluation approach grounded in penetration testing practices. The objective is not limited to identifying potential vulnerabilities, but

extends to assessing their practical exploitability under controlled conditions. In contrast to purely diagnostic methods, penetration testing provides a more realistic perspective on system security by simulating adversarial behavior and observing how the system responds (Wen & Katt, 2023).

The assessment is structured using the Information Systems Security Assessment Framework (ISSAF), which offers a systematic and phased methodology for evaluating information system security. ISSAF is particularly suitable for this study because it integrates both reconnaissance and exploitation stages, allowing the analysis to move beyond theoretical risk toward measurable system resilience. This aligns with recent research emphasizing the importance of combining vulnerability detection with exploit validation to obtain more meaningful security insights (Lachkov et al., 2022).

2.2. Target System and Scope

The target of this study is a publicly accessible web application that functions as a village information system. The platform is designed to provide administrative and informational services to the community, making it a representative example of small-scale public service systems. Due to ethical considerations, the identity of the system—including its domain name and IP address—is partially anonymized throughout this study.

The scope of the assessment is limited to external testing. No direct access to internal infrastructure, server configurations, or administrative interfaces was obtained. This constraint reflects a realistic attacker perspective, where only publicly exposed components are available for interaction. By focusing on the external attack surface, the study aims to evaluate the system's resilience against common web-based threats without interfering with its normal operation.

2.3. Assessment Framework

The security evaluation is conducted using the Information Systems Security Assessment Framework (ISSAF), which structures the assessment into a sequence of logically connected phases. This phased approach ensures that each stage contributes to a comprehensive understanding of the system's security posture.

Unlike unstructured testing methods, ISSAF emphasizes the relationship between different stages of assessment. Information gathered during early phases is used to guide subsequent testing, creating a cumulative analytical process. This approach has been shown to improve the accuracy and effectiveness of penetration testing by reducing randomness and focusing efforts on relevant attack vectors (Auricchio et al., 2022).

The framework applied in this study consists of four main phases: information gathering, network mapping, vulnerability assessment, and penetration testing. Each phase is designed to progressively refine the analysis, moving from general exposure to specific exploitability.

2.4. Tools and Testing Environment

A set of widely recognized security testing tools was used to simulate real-world attack scenarios. These tools were selected based on their reliability, compatibility with web application testing, and relevance to the targeted attack vectors.

- Nmap was used for network scanning and port identification, enabling the analysis of exposed services.
- OWASP ZAP was employed for automated vulnerability scanning, focusing on application-layer weaknesses.
- SQLMap was utilized to test for SQL injection vulnerabilities through automated payload generation.
- Hydra was used to simulate brute-force authentication attacks by attempting multiple credential combinations.
- LOIC (Low Orbit Ion Cannon) was applied to perform controlled denial-of-service simulations.

The testing environment was configured to ensure that all activities were conducted in a controlled and non-destructive manner. All testing activities were conducted in a non-intrusive manner, ensuring that no persistent modifications were introduced to the target web application, and that all tests were performed solely to observe system behavior without causing disruption.

2.5. Assessment Procedure

The assessment procedure follows a sequential workflow aligned with the ISSAF framework. Each phase builds upon the findings of the previous stage, forming a coherent and progressive evaluation process. The overall assessment workflow, including all sequential phases, is illustrated in Figure 1.

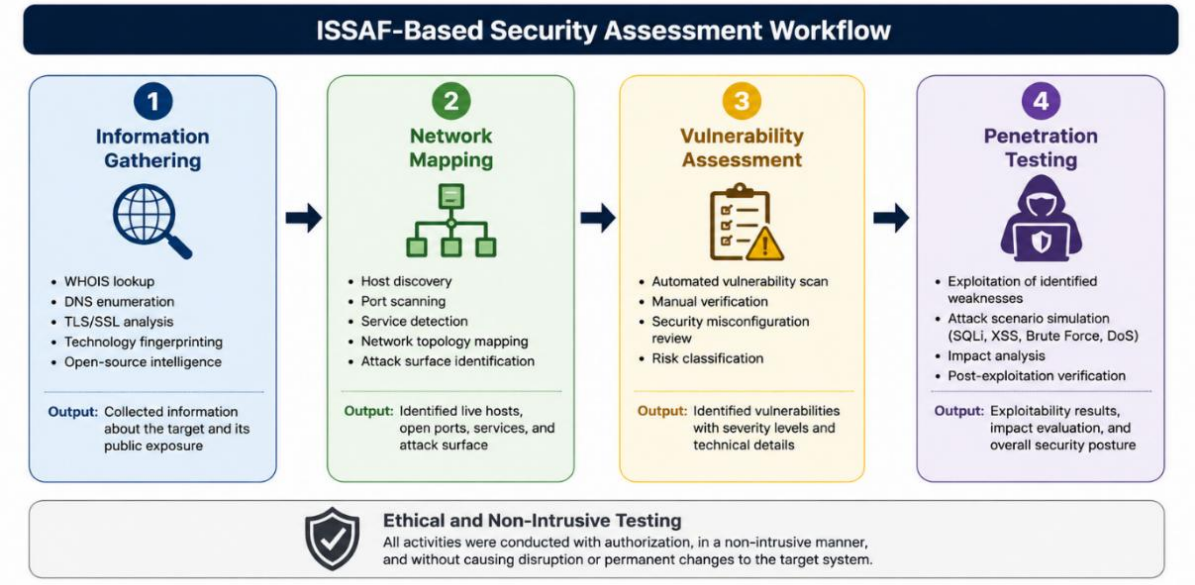

Figure 1. ISSAF-Based Security Assessment Workflow. Sequential phases of the security assessment applied in this study.

2.5.1. Information Gathering

The first phase focuses on collecting publicly accessible information related to the target system. This includes domain resolution, WHOIS data, SSL/TLS configuration, and DNS records. The purpose of this phase is to establish an initial understanding of the system's external exposure without direct interaction with sensitive components.

Previous studies have demonstrated that reconnaissance plays a critical role in shaping attack strategies, as even limited information can significantly improve the efficiency of subsequent exploitation attempts (Lachkov et al., 2022; Wen & Katt, 2023).

2.5.2. Network Mapping

The second phase involves active scanning to identify open ports and accessible services. Nmap is used to probe the system and determine which services are exposed to the public network. The results of this phase define the system's attack surface at the network level.

A limited number of exposed services generally indicates a reduced attack surface, which can lower the likelihood of successful exploitation. However, this does not eliminate risks at the application layer, which require further investigation (Cheng et al., 2017).

2.5.3. Vulnerability Assessment

The vulnerability assessment phase aims to identify weaknesses within the web application. OWASP ZAP is used to detect issues such as missing security headers, misconfigurations, and client-side vulnerabilities. The identified findings are categorized based on severity levels, including medium, low, and informational.

This phase provides a structured overview of potential risks but does not assume that all detected vulnerabilities are exploitable. Instead, it establishes a baseline for further validation through penetration testing.

2.5.4. Penetration Testing

The final phase evaluates the exploitability of the identified vulnerabilities through controlled attack simulations. Several common attack vectors are tested, including:

- SQL injection using SQLMap
- Cross-site scripting (XSS) through manual payload injection
- Brute-force authentication attacks using Hydra
- Denial-of-service simulations using LOIC

This phase shifts the analysis from theoretical risk to actual system behavior. By observing whether vulnerabilities can be successfully exploited, the study provides a more accurate assessment of the system's security resilience (Altulaihan et al., 2023).

2.6. Ethical Considerations

All testing activities were conducted with strict adherence to ethical guidelines. The assessment was designed to avoid any disruption to the system's normal operation. Sensitive information, including IP addresses and administrative details, was partially anonymized to prevent misuse.

Furthermore, the study follows responsible disclosure principles by focusing on analysis rather than exploitation. The goal is to improve system security, not to expose it to additional risk. This approach is consistent with established practices in applied cybersecurity research, where ethical considerations are integral to the assessment process. The results obtained from each phase of the assessment are presented and analyzed in the following section.

3. Results and Discussion

This section presents the results of the security assessment and discusses their implications in relation to the overall security posture of the evaluated web application.

3.1. Information Gathering Results

As illustrated in Figure 2, the initial reconnaissance results provide insight into the external exposure of the web application. The information gathering phase was conducted to establish an initial security profile of the target web application prior to active testing.

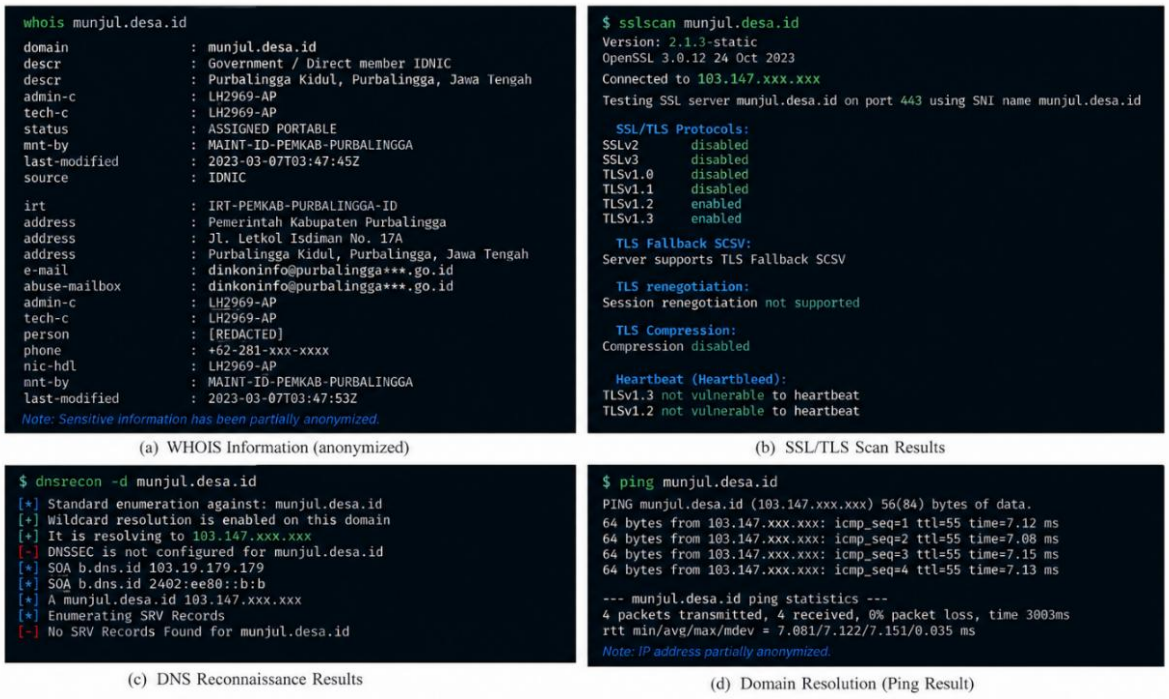

Figure 2. Information Gathering Results of the Target Website. Composite view of the reconnaissance phase, including anonymized WHOIS information, SSL/TLS configuration analysis, DNS reconnaissance findings, and domain resolution results. Sensitive details such as IP addresses, email contacts, and personal identifiers have been partially masked to ensure ethical compliance and prevent misuse, while preserving the technical context necessary for reproducibility.

The information gathering phase was conducted to establish an initial security profile of the target web application prior to active testing. This stage focused on extracting publicly accessible data and configuration details that could inform potential attack vectors. As no direct interaction with internal components was performed, the analysis relies exclusively on externally observable information, which provides valuable insight into both technical exposure and operational practices.

As shown in Figure 2(d), the domain resolution process confirmed that the web application is hosted on a publicly reachable infrastructure. The web application was successfully mapped to a public IP address; however, in line with ethical research practices, the exact address is presented in a masked format (e.g., 103.147.xxx.xxx) to prevent misuse. This approach aligns with responsible disclosure principles, where sensitive identifiers are partially anonymized to balance reproducibility and security

(Cheng et al., 2017). The successful resolution indicates stable network availability, which is essential not only for legitimate access but also for potential adversarial interaction.

Further analysis using WHOIS queries revealed domain registration metadata, including administrative and organizational attributes. As illustrated in Figure 2(a), these details were partially exposed, though sensitive fields such as email addresses and contact numbers have been intentionally redacted in the published output. While WHOIS data is inherently public, its aggregation can facilitate reconnaissance-driven attacks, particularly social engineering and phishing campaigns targeting system administrators. Recent studies highlight that attackers increasingly rely on open-source intelligence (OSINT), including domain metadata, to construct targeted intrusion strategies (Lachkov et al., 2022). In this context, the exposure does not constitute a direct web application vulnerability, yet it expands the human-centric attack surface, which is often less protected than the technical layer.

The transport-layer security configuration was evaluated through SSL/TLS scanning. As shown in Figure 2(b), the system supports both TLS 1.2 and TLS 1.3, indicating adherence to contemporary encryption standards. This observation indicates a positive security posture at the transport layer. TLS 1.3, in particular, reduces handshake latency and eliminates several legacy cryptographic weaknesses present in earlier versions, thereby enhancing both performance and security (Holz et al., 2016). The presence of these protocols suggests that data exchanged between users and the server is adequately protected against interception and passive monitoring attacks.

However, it is important to interpret this result cautiously. The mere availability of modern TLS versions does not guarantee complete transport security. Misconfigurations—such as weak cipher suites or missing enforcement mechanisms—can still introduce vulnerabilities. In this case, no immediate cryptographic weaknesses were observed, suggesting that the implementation is functionally sound within the scope of the assessment.

A more critical observation emerged from the DNS reconnaissance process. As presented in Figure 2(c), the domain does not implement DNS Security Extensions (DNSSEC). The absence of DNSSEC implies that the integrity of DNS responses cannot be cryptographically verified, leaving the system potentially susceptible to DNS spoofing or cache poisoning attacks. Such attacks could redirect users to malicious replicas without altering the visible domain name, making them particularly deceptive (Zhang et al., 2021).

This limitation is especially relevant for public-facing government systems, where user trust is often implicit. Without DNSSEC, the authenticity of domain resolution relies entirely on external infrastructure, which may be exploited under certain threat conditions. Although exploitation is not trivial, the lack of DNSSEC removes an important defensive layer that could otherwise mitigate these risks.

Overall, the findings from the information gathering phase indicate that the web application demonstrates a moderate level of security awareness, particularly in its adoption of modern TLS protocols and controlled network exposure. At the same time, indirect risk factors persist—most notably through publicly available domain

metadata and the absence of DNSSEC protection. These issues do not immediately compromise the system, yet they provide valuable entry points for more sophisticated, multi-stage attack scenarios.

From an analytical perspective, this phase does not reveal exploitable vulnerabilities in isolation. Instead, it highlights how low-level information exposure and configuration gaps can collectively shape the attack surface, setting the stage for subsequent assessment phases.

3.2. Network Mapping Analysis

As presented in Figure 3, the network mapping results show that only ports 80 (HTTP) and 443 (HTTPS) are open, indicating minimal exposure. The network mapping phase was conducted to identify accessible services and evaluate the external attack surface of the web application.

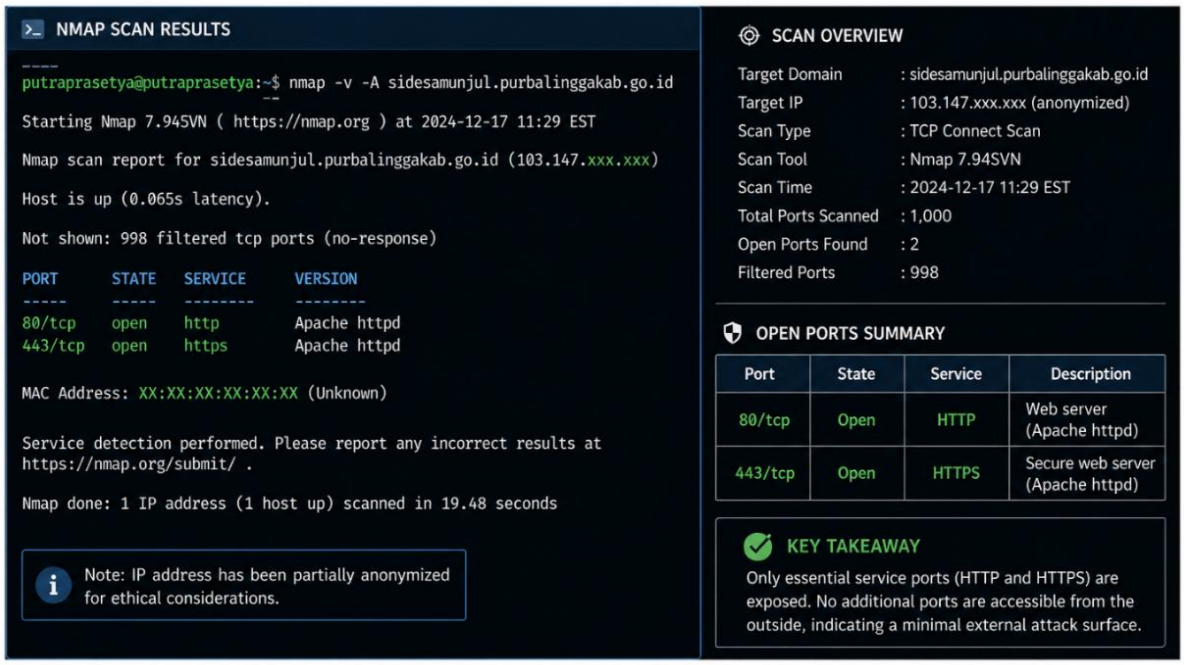

Figure 3. Network Mapping and Open Port Identification. Results of port scanning using Nmap, indicating that only essential service ports (HTTP and HTTPS) are exposed. The IP address has been partially anonymized for ethical considerations

The network mapping phase was conducted to identify accessible services and evaluate the external attack surface of the target web application. This step is critical within the ISSAF framework, as exposed ports and running services often define the initial entry points for exploitation. Unlike the previous phase, which focused on passive data collection, network mapping introduces a more active probing approach—systematically examining how the target responds to connection attempts across a range of ports.

The scan results indicate that only two ports are open and responsive: port 80 (HTTP) and port 443 (HTTPS). No additional ports were detected as accessible during the assessment. Although this configuration may appear straightforward, it reflects a deliberate restriction of unnecessary network exposure. Yet, in practice, it reflects a deliberate—or at least effective—restriction of unnecessary network exposure.

From a security standpoint, limiting open ports to essential services significantly reduces the number of potential attack vectors. Each open port represents a possible interaction channel, and by extension, a potential vulnerability. Therefore, minimizing exposed services is widely recognized as a foundational principle in secure system design (Behl & Behl, 2021). The observed configuration suggests that the web application adheres to this principle, either through proper firewall rules, service hardening, or default hosting configurations.

The presence of HTTP (port 80) alongside HTTPS (port 443), however, introduces a subtle point of discussion. While HTTP is commonly maintained for backward compatibility or automatic redirection, it inherently lacks encryption. If not properly configured to enforce redirection to HTTPS, it may expose users to risks such as man-in-the-middle (MITM) attacks or session interception. Modern best practices strongly recommend either disabling HTTP entirely or ensuring strict redirection policies, ideally reinforced with mechanisms such as HTTP Strict Transport Security (HSTS) (Alashwali et al., 2020).

In this case, although the scan confirms the availability of both ports, the absence of further vulnerabilities in later testing phases suggests that HTTP is likely configured in a controlled manner. Still, without explicit verification of redirection enforcement, this remains a minor area of concern.

Another important observation lies in what is *not* present. No ports associated with administrative services—such as SSH (22), FTP (21), or database services (e.g., MySQL on 3306)—were found to be publicly accessible. This absence is significant. Exposed administrative ports are frequently targeted in automated attacks, including brute-force login attempts and service-specific exploits. Their closure indicates that either such services are disabled externally or restricted to internal networks, both of which are considered strong defensive practices (Sasaki et al., 2022).

The implications of this minimal exposure become more evident when correlated with the results of the penetration testing phase. The inability to exploit the system through common attack vectors—such as brute force or service-based intrusion—can be partially attributed to the lack of accessible entry points at the network level. In other words, the web application does not merely rely on application-layer defenses; it also benefits from a constrained and well-defined network boundary.

Nevertheless, a reduced attack surface does not inherently guarantee comprehensive security. Attackers often shift focus toward application-layer vulnerabilities when network-level entry points are restricted. Therefore, while the network configuration reduces exposure, it does not eliminate risk. Instead, it shifts the defensive burden toward secure application design and proper configuration management.

Overall, the network mapping results demonstrate a controlled and minimalistic exposure model, which aligns with established cybersecurity best practices. The web application effectively limits unnecessary services, thereby reducing opportunities for exploitation. This configuration plays a crucial role in enhancing the system's resilience, particularly against automated and opportunistic

attacks. However, continued attention to service configuration—especially regarding HTTP usage—remains necessary to ensure comprehensive protection.

3.3. Vulnerability Assessment

The identified vulnerabilities are summarized in Table 1, which presents their classification based on severity levels.s

Table 1. Vulnerability Classification Result

Vulnerability Type	Severity Level
Content Security Policy not set	Medium
Cross-domain misconfiguration	Medium
Missing anti-clickjacking header	Medium
Vulnerable JavaScript library	Medium
Cookie without HttpOnly flag	Low
Cookie without Secure flag	Low
Cross-domain JS inclusion	Low
Information leakage (header)	Low
Missing HSTS header	Low
X-Content-Type-Options missing	Low
Informational findings	Informational

As shown in Table 1, the majority of findings fall under medium and low severity categories, primarily related to configuration weaknesses rather than critical exploitable flaws.

As illustrated in Figure 4, the majority of identified vulnerabilities fall within the low and medium severity categories, with low-level issues accounting for the largest proportion. This distribution indicates that while no critical vulnerabilities were detected, the system still contains several configuration-related weaknesses that may contribute to increased security risk if not addressed.

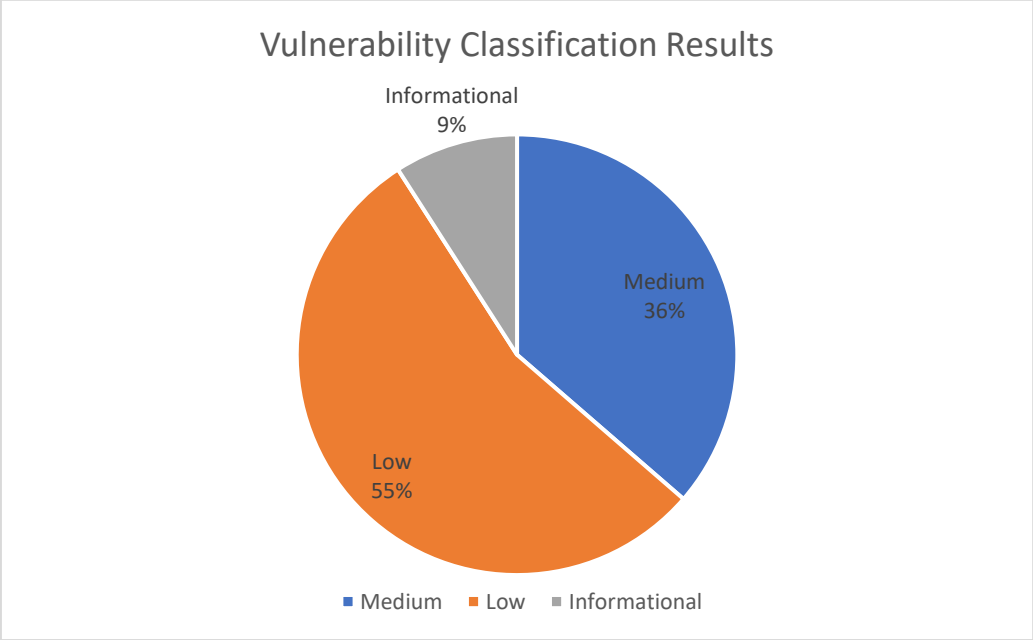

Figure 4. Vulnerability Severity Distribution. Distribution of identified vulnerabilities based on severity levels. The results indicate that most findings fall under low and medium categories, with no critical vulnerabilities detected

The vulnerability assessment phase aimed to identify weaknesses within the application layer that could potentially be leveraged by an attacker as widely discussed in prior web security studies (A. Correa et al., 2021). Using automated scanning tools complemented by manual inspection, the analysis revealed a total of four medium-level vulnerabilities, six low-level vulnerabilities, and several informational findings. The absence of high or critical vulnerabilities initially suggests a relatively secure posture. However, a closer examination reveals a more nuanced reality—one where risk is distributed across configuration gaps rather than concentrated in a single exploitable flaw.

Rather than treating these findings as isolated issues, it is more meaningful to interpret them as part of a broader security posture. The detected vulnerabilities can be grouped into three primary categories: security misconfigurations, missing security headers, and client-side weaknesses.

3.3.1. Security Misconfigurations and Header Weaknesses

A significant portion of the identified issues falls under missing or improperly configured HTTP security headers, including the absence of Content Security Policy (CSP) and X-Frame-Options. These headers are not merely optional enhancements; they function as critical safeguards against common web-based attacks.

The absence of CSP, for instance, does not immediately allow code execution. However, it removes an essential layer of defense against injection-based attacks, particularly Cross-Site Scripting (XSS). Without CSP, the browser lacks explicit instructions on which sources are trusted, thereby increasing the likelihood that malicious scripts—if introduced—could execute without restriction (Roth et al., 2020).

Similarly, the lack of X-Frame-Options exposes the application to clickjacking attacks, where malicious actors can embed the website within invisible frames to trick users into unintended interactions. While such attacks require user involvement, their effectiveness has been demonstrated in various real-world scenarios, particularly in systems with insufficient UI protection (Calzavara et al., 2020).

When considered individually, these issues may appear minor. Collectively, however, they signal a systematic gap in defensive configuration practices.

3.3.2. Client-Side and Session-Related Weaknesses

Another set of findings relates to cookie security attributes, specifically the absence of HttpOnly and Secure flags. These flags play a crucial role in protecting session data. Without the HttpOnly attribute, cookies can be accessed via client-side scripts, making them vulnerable in the presence of XSS vulnerabilities. Likewise, the absence of the Secure flag allows cookies to be transmitted over unencrypted connections, potentially exposing them to interception.

Although no active XSS vulnerability was successfully exploited in subsequent testing, the presence of weak cookie configurations increases the impact severity should such an attack become feasible. In other words, these weaknesses do not create an attack—but they amplify its consequences.

Additionally, the use of outdated or vulnerable JavaScript libraries was identified. This represents a commonly observed yet frequently underestimated issue in web application security. Legacy libraries may contain known vulnerabilities that can be exploited through publicly available attack vectors. Studies have shown that a

significant percentage of web applications continue to rely on outdated dependencies, thereby inheriting their associated risks (Berry, 2021).

In this context, the risk is not immediate exploitation, but rather latent exposure—a vulnerability waiting for the right conditions.

3.3.3. Informational Findings and Information Disclosure

The informational findings, while not directly exploitable, provide insight into the system's internal structure and behavior. These include server response headers disclosing technology stacks and the presence of suspicious comments within source code.

Such disclosures may appear harmless. Yet, from an attacker's perspective, they function as reconnaissance accelerators. By identifying the technologies in use, attackers can narrow down potential exploits and tailor their approach accordingly. This aligns with the concept of information leakage as a precursor to targeted attacks, where even minor details contribute to a more efficient exploitation strategy (Sabottke et al., n.d.).

3.3.4. Risk Interpretation and Security Implications

What makes these findings particularly interesting is not their individual severity, but their collective implication. None of the identified vulnerabilities provide direct system access. There is no immediate pathway to database extraction, remote code execution, or privilege escalation.

And yet, the web application cannot be considered fully secure.

The vulnerabilities identified in this phase primarily function as risk multipliers. They do not initiate attacks, but they lower the barrier for successful exploitation when combined with other weaknesses. This layered risk model is consistent with modern cybersecurity threat landscapes, where attacks rarely rely on a single point of failure but instead exploit chains of smaller weaknesses (Allodi et al., 2022).

In practical terms, this means that the web application demonstrates resistance to direct attacks, but remains susceptible to more sophisticated, multi-stage exploitation scenarios. The absence of critical vulnerabilities should not lead to complacency. On the contrary, it highlights the importance of addressing configuration-level issues before they evolve into exploitable conditions.

3.3.5. Overall Assessment

The vulnerability assessment reveals a system that is structurally secure but operationally imperfect. Its defenses are sufficient to prevent immediate compromise, yet not comprehensive enough to eliminate all potential risks.

This highlights the importance of addressing configuration-level weaknesses in maintaining overall security.

3.4. Penetration Testing Evaluation

As illustrated in Figure 5 the penetration testing phase includes multiple attack scenarios, all of which resulted in unsuccessful exploitation attempts.

Figure 5. Penetration Testing Results Across Multiple Attack Scenarios. Summary of controlled attack simulations including SQL injection, cross-site scripting (XSS), brute-force authentication attempts, and denial-of-service (DoS) testing. All attack attempts were unsuccessful, indicating the system's resilience against common web-based exploitation techniques.

The penetration testing phase represents the culmination of the ISSAF assessment, where previously identified weaknesses are actively tested to determine their exploitability as a standard approach in practical security evaluation (Lachkov et al., 2022). Unlike earlier stages—focused on exposure and configuration—this phase attempts to answer a more direct question: *can the web application actually be compromised?*

A series of controlled attack simulations were conducted, targeting common and high-impact web vulnerabilities. These included SQL injection, cross-site scripting (XSS), brute-force authentication attacks, and denial-of-service (DoS) scenarios. The selection reflects widely observed attack vectors in real-world web environments, particularly in public-facing systems (Altulaihan et al., 2023).

3.4.1. SQL Injection Testing

The system was first evaluated for susceptibility to SQL injection using automated tools designed to identify improper input handling and database query manipulation. The testing process involved injecting crafted payloads into input fields and observing server responses for anomalies or unintended behavior.

The results indicate that no SQL injection vulnerability was detected. The application consistently sanitized or rejected malicious input, preventing any form of query manipulation. This suggests that input validation mechanisms and backend query handling are implemented with a reasonable level of security awareness.

From a broader perspective, this finding is significant. SQL injection remains one of the most critical web vulnerabilities, often leading to full database compromise when present. Its absence here implies that the application has effectively mitigated one of the most commonly exploited attack vectors (Disawal & Suman, 2023).

3.4.2. Cross-Site Scripting (XSS) Evaluation

XSS testing was conducted by injecting script-based payloads into various input vectors, including URL parameters and form fields. The objective was to determine whether the application improperly reflects or stores user-supplied input in a way that allows script execution within the browser.

The web application demonstrated no successful execution of injected scripts, indicating resistance to both reflected and basic stored XSS attacks. This suggests that output encoding and input filtering mechanisms are functioning correctly.

However, it is important to interpret this result in conjunction with earlier findings. The absence of a Content Security Policy (CSP), as identified in Section 3.3, means that while XSS is currently not exploitable, the system lacks a secondary defense layer. Should an injection vector emerge in the future, the impact could be significantly higher. This highlights a recurring theme: security is not only about preventing vulnerabilities, but also about limiting their consequences.

3.4.3. Brute-Force Attack Simulation

Authentication mechanisms were tested against brute-force attacks using automated credential guessing techniques. These attacks attempt to gain unauthorized access by systematically trying multiple username-password combinations.

The results show that no unauthorized access was achieved, even under repeated login attempts. This suggests the presence of protective measures such as rate limiting, account lockout policies, or sufficiently strong authentication controls.

This outcome is particularly relevant given the prevalence of credential-based attacks in modern threat landscapes. Studies indicate that brute-force and credential stuffing attacks remain among the most frequently observed intrusion techniques, especially in web applications lacking adequate authentication defenses (García et al., 2014). The system's resistance in this area reflects a meaningful level of maturity in access control implementation.

3.4.4. Denial-of-Service (DoS) Testing

A controlled DoS simulation was performed using traffic generation tools to evaluate the system's resilience under high request volumes. The objective was not to disrupt the service permanently, but to observe its behavior under stress conditions.

The web application remained accessible and responsive during the test, indicating a basic level of resilience against volumetric attacks. While this does not guarantee protection against large-scale distributed denial-of-service (DDoS) attacks, it suggests that the underlying infrastructure can handle moderate traffic spikes without immediate degradation.

It is worth noting that DoS resistance is often influenced by external factors such as hosting infrastructure, load balancing, and network-level protections. Therefore, this result should be interpreted as indicative rather than definitive.

3.4.5. Integrated Security Interpretation

Taken individually, each test result points toward a system that is resistant to direct exploitation. No injection points were successfully leveraged, no unauthorized access was obtained, and no service disruption was observed under controlled

conditions. A more critical insight emerges when these results are analyzed collectively.

Despite the presence of several vulnerabilities identified in Section 3.3—particularly related to misconfigurations and missing security headers—none of them translated into practical exploitation paths. This suggests that the system benefits from a form of defense-in-depth, where multiple layers of protection compensate for individual weaknesses (Altulaihan et al., 2023).

At the same time, this does not imply absolute security. The vulnerabilities identified earlier still represent latent risks, which may become exploitable under different conditions or in combination with new attack techniques. Modern cyberattacks increasingly rely on chaining multiple low-severity issues rather than exploiting a single critical flaw (Zhang et al., 2021).

In this context, the web application can be characterized as resilient but not immune. It withstands common attack scenarios effectively, yet still requires proactive hardening to address configuration-level weaknesses.

A summary of the penetration testing outcomes is presented in Table 2, highlighting the system's resistance across different attack vectors.

Table 2. Penetration Testing Summary

Attack Type	Result	Status
SQL Injection	Not detected	Secure
XSS	Not executed	Secure
Brute Force	Failed	Secure
DoS	No disruption	Resilient

3.4.6. Overall Evaluation

The penetration testing results reinforce the conclusion that the system maintains a moderate-to-strong security posture against widely known attack vectors. Its ability to resist exploitation across multiple scenarios indicates that fundamental security controls are in place and functioning as intended.

However, the absence of successful attacks should not be interpreted as the absence of risk. Security, in practice, is a moving target. What cannot be exploited today may become viable tomorrow—especially in environments where minor weaknesses remain unaddressed.

4. Conclusion

This study evaluated the security posture of a public village information system using the Information Systems Security Assessment Framework (ISSAF) by combining vulnerability assessment with penetration testing to examine both the existence and exploitability of security weaknesses. The findings indicate that the web application has a moderate level of security. Although reconnaissance identified publicly accessible metadata and the absence of DNSSEC, and the vulnerability assessment revealed several low- and medium-severity misconfigurations such as missing security headers, weak cookie settings, and outdated client-side components, no critical vulnerabilities were detected. Network mapping also showed a limited attack surface with only essential ports exposed. Furthermore, penetration testing

using SQL injection, cross-site scripting (XSS), brute-force authentication, and denial-of-service scenarios did not result in successful exploitation, indicating that key security mechanisms, including input validation and access control, are functioning effectively. Nevertheless, the identified configuration weaknesses may still increase the risk of multi-stage attacks if left unaddressed, suggesting that the system is resilient but not completely immune to evolving cyber threats.

The results demonstrate that ISSAF provides a practical and systematic approach for assessing the security of small-scale public web applications, particularly in resource-constrained e-government environments. Beyond identifying vulnerabilities, the framework enables organizations to evaluate the actual impact of security weaknesses through controlled exploitation, supporting more effective risk mitigation. Future research may extend this work by evaluating multiple public service systems, incorporating continuous security monitoring, or comparing ISSAF with other security assessment frameworks to obtain a broader understanding of web application security in the public sector.

Acknowledgement

The authors would like to express their sincere appreciation to their respective institutions for providing academic support and access to research resources that contributed to the completion of this study. The authors are also grateful to the reviewers and editors for their valuable comments and constructive suggestions that helped improve the quality of this manuscript.

Author Contributions

Putra Prasetya, Harjono: Conceptualization, Methodology, Data collection, Formal analysis, Writing—original draft, Writing—review and editing.
Mukhlis Prasetyo Aji, Ermadi Satriya Wijaya: Conceptualization, Methodology, Visualization, Writing—review and editing, Supervision.

Funding

This research received no external funding.

Declaration on the Use of Artificial Intelligence

Artificial intelligence support was used in a limited capacity through ChatGPT (OpenAI), based on the GPT-5 family model, solely for language improvement, including grammar correction, sentence refinement, clarity enhancement, and overall readability of the manuscript. The AI tool was not used for data collection, data analysis, interpretation of results, generation of findings, reference selection, or scholarly decision-making. All research activities, intellectual contributions, methodological design, critical interpretation, and final manuscript approval were carried out entirely by the authors. The authors take full responsibility for the accuracy, originality, and integrity of the published work.

References

A. Correa, R., Ram髇 Bermejo Higuera, J., Bermejo Higuera, J., Antonio SiciliaMontalvo, J., S醬chez Rubio, M., & Alberto Magre襻n, □. (2021). Hybrid Security AssessmentMethodology forWeb Applications. *Computer Modeling in Engineering & Sciences*, *126*(1), 89–124. https://doi.org/10.32604/cmes.2021.010700

Alashwali, E. S., Szalachowski, P., & Martin, A. (2020). Exploring HTTPS security inconsistencies: A cross-regional perspective. *Computers & Security*, *97*, 101975. https://doi.org/10.1016/j.cose.2020.101975

Allodi, L., Massacci, F., & Williams, J. (2022). The Work-Averse Cyberattacker Model: Theory and Evidence from Two Million Attack Signatures. *Risk Analysis*, *42*(8), 1623–1642. https://doi.org/10.1111/risa.13732

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

Auricchio, N., Cappuccio, A., Caturano, F., Perrone, G., & Romano, S. Pietro. (2022). An automated approach to Web Offensive Security. *Computer Communications*, *195*, 248–261. https://doi.org/10.1016/j.comcom.2022.08.018

B, R. K., Godishala, A. K., Malaga, R. R., & Kagitapu, P. (2024). Securing web apps: Analysis to understand common vulnerabilities, attack scenarios, and protective measures. *Proceedings of the 2024 10th International Conference on Computing and Data Engineering*, 64–70. https://doi.org/10.1145/3641181.3641187

Berry, D. M. (2021). Empirical evaluation of tools for hairy requirements engineering tasks. *Empirical Software Engineering*, *26*(6), 111. https://doi.org/10.1007/s10664-021-09986-0

Cheng, L., Liu, F., & Yao, D. (Daphne). (2017). Enterprise data breach: causes, challenges, prevention, and future directions. *WIREs Data Mining and Knowledge Discovery*, *7*(5). https://doi.org/10.1002/widm.1211

Darojat, E. Z., Sediyono, E., & Sembiring, I. (2022). Vulnerability Assessment Website E-Government dengan NIST SP 800-115 dan OWASP Menggunakan Web Vulnerability Scanner. *JURNAL SISTEM INFORMASI BISNIS*, *12*(1), 36–44. https://doi.org/10.21456/vol12iss1pp36-44

Disawal, S., & Suman, U. (2023). An Approach to Detect and Prevent SQL Injection and XSS Vulnerability in the Web Application. *International Journal of Engineering Trends and Technology*, *71*(8), 216–224. https://doi.org/10.14445/22315381/IJETT-V71I8P219

García, S., Grill, M., Stiborek, J., & Zunino, A. (2014). An empirical comparison of botnet detection methods. *Computers & Security*, *45*, 100–123. https://doi.org/10.1016/j.cose.2014.05.011

Holz, R., Amann, J., Mehani, O., Wachs, M., & Kaafar, M. A. (2016). TLS in the Wild: An Internet-wide Analysis of TLS-based Protocols for Electronic Communication. *23rd Annual Network and Distributed System Security Symposium, NDSS 2016*. https://doi.org/10.14722/ndss.2016.23055

Lachkov, P., Tawalbeh, L., & Bhatt, S. (2022). Vulnerability Assessment for Applications Security Through Penetration Simulation and Testing. *Journal of Web Engineering*. https://doi.org/10.13052/jwe1540-9589.2178

Roth, S., Barron, T., Calzavara, S., Nikiforakis, N., & Stock, B. (2020). Complex Security Policy? A Longitudinal Analysis of Deployed Content Security Policies. *Proceedings 2020 Network and Distributed System Security Symposium*. https://doi.org/10.14722/ndss.2020.23046

Sabottke, C., Suciu, O., Dumitraş, T., & Dumitras, T. (n.d.). *Vulnerability Disclosure in the Age of Social Media: Exploiting Twitter for Predicting Real-World Exploits*. Retrieved https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/sabottke

Sasaki, T., Fujita, A., Gañán, C. H., van Eeten, M., Yoshioka, K., & Matsumoto, T. (2022). Exposed Infrastructures: Discovery, Attacks and Remediation of Insecure ICS Remote Management Devices. *2022 IEEE Symposium on Security and Privacy (SP)*, 2379–2396. https://doi.org/10.1109/SP46214.2022.9833730

Wen, S.-F., & Katt, B. (2023). A quantitative security evaluation and analysis model for web applications based on OWASP application security verification standard. *Computers & Security*, *135*, 103532. https://doi.org/10.1016/j.cose.2023.103532

Zhang, H., Ye, J., Hu, W., Wang, Q., Yan, X., Yue, Q., Lv, W., He, M., & Wang, J. (2021). Study on the latent state of Kaminsky-style DNS cache poisoning: Modeling and empirical analysis. *Computers & Security*, *110*, 102445. https://doi.org/10.1016/j.cose.2021.102445