BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Yang Relevan

Penelitian yang relevan digunakan sebagai pembanding penelitian-penelitian sebelumnya yang akan memperkuat posisi penelitian ini.

Tabel 2. 1 Penelitian Relevan

No	**1**
Judul Penelitian	Analisis Keamanan *Web Server Open Journal System* (OJS) Menggunakan Metode ISSAF dan OWASP (Studi Kasus OJS Universitas Lancang Kuning)
Penulis	Guntoro, Loneli Costaner, Musfawati
Tahun	2020
Hasil	Pengujian penetrasi yang di bahas menggunakan ISSAF dan OWASP versi 4 menunjukkan bahwa sistem OJS Universitas Lancang Kuning secara umum aman dari eksploitasi yang dilakukan dari pihak eksternal.Namun ditemukan kelemahan pada mekanisme lockout login(tidak memblokir percobaan login gagal berulang), dan juga sistem rentan terhadap serangan DoS (Denial of Service) tetapi tidak terhadap serangan SQL injection. adapun rekomendasi dari peneliti untuk perbaikan adalah mencakup penerapan IDS (Intrusion Detection System), backup data secara berkala, update sistem OJS (Open Journal System) ke yang lebih terbaru secara berkala, dan dapat menerapkan pemblokiran terhadap login yang tidak valid sesuai pada tahapan Testing for Weak lock out mechanism (OTG-AUTHN-003).
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menguji keamanan sistem Open Journal System (OJS)

	menggunakan pendekatan pengujian keamanan standar internasional. Perbedaannya ialah penelitian tersebut hanya menguji OJS dalam arsitektur monolitik.
No	2
Judul	A Systematic Study of Network Firewall and Its Implementation
Penulis	Raed Alsaqour, Ahmed Motmi, Maha Abdelhaq
Tahun	2021
Hasil	Penelitian ini membahas berbagai jenis firewall (packet filtering, stateful inspection, proxy, next-generation, arsitekturnya, mekanisme kerja, serta kerentanan umum seperti kesalahan konfigurasi, serangan DDoS, dan ancaman internal). Firewall terbukti efektif sebagai lapisan pertama pertahanan jaringan tetapi tidak cukup jika berdiri sendiri perlu dikombinasikan dengan mekanisme keamanan lain seperti IDS/IPS dan segmentasi jaringan.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menggunakan firewall sebagai lapisan pertama pertahanan jaringan. Perbedaannya ialah penelitian tersebut hanya membahas firewall secara umum.
No	3
Judul	Virtual Local Area Network (VLAN) Network Design for NEMSU-Administration Building
Penulis	Emmer P. Ruaya, Mark Van M. Buladaco
Tahun	2022

Hasil	Penelitian ini merancang dan mengimplementasikan VLAN(Virtual Local Area Network) pada gedung administrasi North Eastern Mindanao State University (NEMSU) untuk memisahkan lalu lintas jaringan berdasarkan departemen contoh (Keuangan, HRD, Direktorat) Hasilnya menunjukkan bahwa VLAN efektif dalam meningkatkan keamanan internal, mengurangi broadcast traffic, mempermudah manajemen jaringan, serta meningkatkan Qualitiy of Service (QoS). Keamanan diperkuat dengan firewall dan pembatasan akses berdasarkan peran.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menerapkan prinsip segregasi jaringan untuk meningkatkan keamanan internal. Perbedaannya ialah penelitian tersebut menggunakan VLAN untuk memisahkan departemen.
No	4
Judul	Safety Protection Design of Virtual Machine Drift Flow in Cloud Data Center Based on VXLAN Technology
Penulis	Heping Pu, Yijun Wang, Xinke An
Tahun	2020
Hasil	Penelitian yang dilakukan yaitu mengusulkan arsitektur keamanan berbasis virtualisasi untuk pusat data cloud dengan memanfaatkan teknologi VXLAN untuk mengisolasi lalu lintas horizontal antar VM semua lalu lintas antar-VM (east-west traffic) dialihkan ke security gateway eksternal yang terpusat (menggunakan Huawei NSH service chaining) sehingga memungkinkan penerapan kontrol keamanan granular atau keamanan

	yang detail dan spesifik tanpa mengorbankan performa VM. Skema ini efektif mencegah penyebaran ancaman lateral contoh (virus,eksploitasi) antar VM dan mendukung perpindahan VM yang aman.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menggunakan isolasi VM untuk mencegah penyebaran serangan lateral. Perbedaannya ialah penelitian tersebut menggunakan VXLAN dan gateway keamanan enterprise.
No	5
Judul	Analisis Keamanan Pada Web Aplikasi Open Journal System Terhadap Serangan Cross Site Scripting (XSS) Menggunakan Metode Vulnerability Assessment
Penulis	Yunanri W
Tahun	2023
Hasil	Pengujian menggunakan OWASP ZAP terhadap OJS Jurnal Informatika Teknologi dan Sains (JINTEKS) mengidentifikasi 12 jenis kerentanan yaitu 2 berisiko medium, 7low, dan 3 bersifat informational.Namun tidak ditemukan kerentanan XSS aktif baik stored maupun reflected. Tetapi ditemukan isu keamanan seperti missing security headers (X-Frame-Options, X-Content-Type-Options), cookie tanpa atribut HttpOnly/SameSite, dan pengungkapan informasi sensitif.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menguji kerentanan XSS pada OJS menggunakan OWASP ZAP. Perbedaannya ialah penelitian tersebut hanya mendeteksi kerentanan dalam satu server monolitik.

No	6
Judul	Server virtualization in higher educational institutions: a case study
Penulis	Sreelakshmi Koratagere, Ravi Kumar Chandrashekarappa Koppal, Iyyanahalli Math Umesh
Tahun	2023
Hasil	Studi kasus di RV College of Engineering menunjukkan bahwa migrasi dari 36 server fisik ke 3 server virtual menggunakan infrastruktur hyper-converged berhasil mengurangi konsumsi daya dari 5760 W menjadi 1275 W meningkatkan pemanfaatan sumber daya hardware, mempermudah pemeliharaan, dan memungkinkan penyediaan VM baru dalam hitungan menit keamanan data juga meningkat karena semua layanan terkonsentrasi di pusat data yang terlindungi.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menerapkan virtualisasi server di lingkungan pendidikan tinggi untuk efisiensi. Perbedaannya ialah penelitian tersebut fokus pada penghematan biaya dan manajemen.
No	7
Judul	Analisis Keamanan *Website Open Journal System* Menggunakan Metode *Vulnerability Assessment*
Penulis	Imam Riadi, Anton Yudhana, Yunanri W.
Tahun	2020
Hasil	Pengujian terhadap OJS versi 2.4.7 menggunakan OWASP ZAP mengidentifikasi 6.049 kerentanan: 70high, 1.929medium, dan 4.050low. Kerentanan utama

	meliputi SQL Injection,Cross-Site Scripting (XSS),missing security headers(X-Frame-Options, X-Content-Type-Options), dan cookie tanpa HttpOnly. Studi ini menyimpulkan bahwa OJS versi lama sangat rentan dan tidak direkomendasikan untuk digunakan tanpa pembaruan ke versi terbaru dari PKP.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya mengidentifikasi kerentanan SQLi dan XSS pada OJS versi lama. Perbedaannya ialah penelitian tersebut hanya melaporkan kerentanan.
No	8
Judul	*Studi dan Analisis Keamanan Sistem Internal Docker dari Docker Daemon Attack dan DDoS pada Sistem Open Journal System*
Penulis	Choirul Rio Prabowo, Dodo Irmanto, Erfan Rohadi
Tahun	2024
Hasil	Penelitian ini menguji ketahanan sistem OJS yang dijalankan di atas Docker container terhadap dua ancaman yaitu Docker Daemon Attack akibat misconfiguration dan serangan DDoS. Hasil menunjukkan bahwa dengan konfigurasi keamanan yang tepat seperti pembatasan akses ke Docker socket dan penerapan DDoS mitigation berbasis sFlow/ExaBGP sistem Docker mampu menahan kedua jenis serangan tersebut tetapi sangat ditekankan bahwa kesalahan konfigurasi container dapat memberikan akses root kepada attacker yang berpotensi mengganggu seluruh sistem.

Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya mengamankan OJS dalam lingkungan virtualisasi. Perbedaannya ialah penelitian tersebut menggunakan Docker (container) yang memiliki isolasi lebih lemah.
No	9
Judul	*Web Server Analysis for Open Journal System Needs Using Secure Tunnel*
Penulis	Yudhi Arta, Rizky Wandri, Anggi Hanafiah, Bima Kristian Pranoto, M Rizki Fadhilah
Tahun	2022
Hasil	Di dalam penelitian ini membandingkan kinerja Apache dan Nginx sebagai web server untuk OJS di bawah beban tinggi (500–1000) pengguna simultan menggunakan stress test dengan Apache JMeter. Hasil menunjukkan bahwa Apache lebih unggul dalam stabilitas error rate lebih rendah, serta performasent/received lebih konsisten. Namun sebaliknya Nginx memiliki throughput dan response time lebih cepat tetapi mengalami tingkat error sangat tinggi hingga (61,9%) saat diakses melalui secure tunnel bahkan gagal memuat header dan footer OJS.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menguji kemampuan OJS menangani beban tinggi. Perbedaannya ialah penelitian tersebut hanya membandingkan Apache vs Nginx.
No	10
Judul	Perancangan Virtual Server Berbasis Proxmox di Laboratorium Komputer Politeknik Belitung

Penulis	Aldof Billpurta, Aryanda, Eka Indah Wahyuni
Tahun	2025
Hasil	Penelitian ini berhasil merancang infrastruktur virtualisasi berbasis Proxmox VE yang menjalankan beberapa VM (web server, database, dll.) dalam satu server fisik. Sistem ini mengoptimalkan penggunaan sumber daya, mengurangi biaya perangkat keras, dan mempermudah manajemen melalui antarmuka web. Proxmox dipilih karena sifatnya open-source mendukung KVM/LXC (Kernel-based Virtual Machine), serta menyediakan fitur keamanan seperti RBAC (Role-Based Access Control), firewall terintegrasi, isolasi VM, dan autentikasi dua faktor (2FA).
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menggunakan Proxmox sebagai platform virtualisasi utama. Perbedaannya ialah penelitian tersebut fokus pada efisiensi dan manajemen infrastruktur.
No	11
Judul	XSS Injection Vulnerability pada Open Journal Systems (OJS)
Penulis	Ismail Puji Saputra, Arif Hidayat
Tahun	2025
Hasil	Penelitian ini berhasil mengidentifikasi dan mengeksploitasi kerentanan XSS (Cross-Site Scripting) pada OJS versi 3.1.1-4 melalui fitur discussion panel yang aktif saat artikel masuk tahap review. Penyerang dapat menyisipkan skrip berbahaya yang mencuri sesi

	pengelola jurnal memungkinkan akses ilegal ke sistem. Serangan dapat dicegah dengan filtering input pada field diskusi khususnya pada file QueryForm.inc.php.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menguji kerentanan XSS spesifik pada fitur diskusi OJS. Perbedaannya ialah penelitian tersebut hanya memberikan patching aplikasi.
No	12
Judul	Deteksi Serangan Vulnerability Pada Open Journal System Menggunakan Metode Black-Box
Penulis	Yunanri.W, Doddy Teguh Yuwono, Rodianto, Yuliadi
Tahun	2021
Hasil	Pengujian penetrasi di lakukan dengan metode black-box dan tools Vega Framework berhasil mengidentifikasi 98 kerentanan pada OJS 1 high risk (SQL Injection),7 medium risk (local filesystem paths found), dan 90 low risk (directory listing detected). Selain itu ditemukan 1.043 informasi tambahan yang perlu ditindak lanjuti untuk perbaikan keamanan.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya menggunakan metode black-box dan tools seperti Vega/OWASP untuk mendeteksi kerentanan OJS. Perbedaannya ialah penelitian tersebut hanya mengidentifikasi dan melaporkan kerentanan.
No	13
Judul	Threat Detection and Mitigation Techniques in Web Application Security: Systematic Review

Penulis	A. F. Lufna, F. S. Ahamed, M. I. Abzan Begum, M. R. M. Hanan, and A. Mohamed Aslam Sujah
Tahun	2025
Hasil	Penelitian ini membahas deteksi dan mitigasi serangan SQL Injection dan Cross-Site Scripting (XSS). Hasilnya menunjukkan bahwa metode yang paling banyak digunakan adalah kombinasi static analysis, dynamic analysis, penetration testing, AI/ML-based detection, dan penerapan *secure SDLC*. XSS dan SQLi tetap menjadi serangan paling dominan pada aplikasi web. Pendekatan berbasis machine learning menjanjikan namun masih memiliki keterbatasan pada skalabilitas dan implementasi nyata. Sistem keamanan berlapis dan pengujian otomatis terbukti meningkatkan ketahanan aplikasi web.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya membahas serangan XSS sebagai ancaman utama pada aplikasi web serta pentingnya deteksi dan mitigasi XSS melalui pendekatan analisis statis/dinamis dan pengujian keamanan. Penelitian tersebut bersifat *systematic review* (kajian literatur) dan tidak melakukan eksperimen langsung pada OJS atau sistem spesifik melainkan merangkum berbagai metode deteksi dan mitigasi XSS secara umum.
No	14
Judul	Mitigation Web Server for Cross-Site Scripting Attack Using Penetration Testing Method
Penulis	Abdul Fadlil, Imam Riadi, Fahmi Fachri
Tahun	2022

Hasil	Penelitian ini bertujuan memitigasi serangan Cross-Site Scripting (XSS) pada web server dengan menggunakan metode penetration testing. Objek penelitian adalah form login pada web server sistem informasi akademik. Hasil pengujian menemukan tiga kategori kelemahan yaitu 5 kerentanan tingkat tinggi, 164 tingkat menengah, dan 52 tingkat rendah. Upaya mitigasi dilakukan dengan menerapkan secure code seperti penolakan terhadap percobaan login gagal berulang. Metode ini berperan dalam mengidentifikasi celah keamanan dan menyediakan mekanisme pencegahan XSS secara lebih sistematis.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya membahas serangan XSS pada web server dan menggunakan metode penetration testing sebagai pendekatan utama dalam mendeteksi dan memitigasi kerentanan. Penelitian tersebut fokus pada aplikasi sistem informasi akademik secara umum bukan secara spesifik pada platform Open Journal Systems (OJS) dan belum mengaitkan mitigasi XSS dengan arsitektur infrastruktur keamanan berbasis open source.
No	15
Judul	GenXSS: an AI-Driven Framework for Automated Detection of XSS Attacks in WAFs
Penulis	Vahid Babaey, Arun Ravindran
Tahun	2025
Hasil	Penelitian ini mengusulkan framework GenXSS yaitu sistem berbasis Artificial Intelligence (AI) dan Large

	Language Models (LLM) untuk mendeteksi dan memitigasi serangan Cross-Site Scripting (XSS) pada Web Application Firewall (WAF). Framework ini menghasilkan payload XSS yang kompleks dan tervalidasi secara sintaksis menggunakan in-context learning kemudian menguji payload tersebut terhadap aplikasi yang dilindungi WAF (ModSecurity). Hasil eksperimen menunjukkan bahwa dari 264 payload XSS, sebanyak 83% valid, dan 80% berhasil melewati ModSecurity WAF dengan rule standar OWASP. Setelah dilakukan pembuatan rule otomatis 86% serangan yang sebelumnya lolos berhasil diblokir hanya dengan 15 rule baru. Hal ini membuktikan efektivitas pendekatan AI dalam meningkatkan kemampuan WAF dalam mitigasi XSS.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya membahas mitigasi serangan XSS dan penerapan Web Application Firewall (WAF) sebagai bagian dari arsitektur keamanan web. Penelitian tersebut menggunakan pendekatan AI dan LLM untuk otomatisasi deteksi dan pembuatan rule WAF, sedangkan skripsi ini berfokus pada perancangan dan evaluasi arsitektur infrastruktur keamanan berbasis open source untuk mitigasi XSS pada platform OJS dengan pendekatan Peneliti memilih pendekatan LWAF karena lebih ringan (*lightweight*) untuk dijalankan di lingkungan kontainer LXC pada Proxmox VE serta lebih mudah direplikasi oleh institusi akademik yang memiliki keterbatasan sumber daya komputasi. Selain itu jika penelitian tersebut berfokus pada aplikasi web secara umum skripsi ini secara spesifik mengevaluasi

	efektivitas arsitektur keamanan tersebut pada platform OJS versi 3.3.0-16 dan 3.4.0-5.
No	16
Judul	Cross-Site Scripting Attacks and Defensive Techniques: A Comprehensive Survey
Penulis	Sonkarlay J. Y. Weamie
Tahun	2022
Hasil	Penelitian ini merupakan survei komprehensif yang membahas secara mendalam tentang serangan Cross-Site Scripting (XSS), metode deteksi, serta teknik pencegahannya. Penulis mengelompokkan pendekatan pertahanan XSS ke dalam lima kategori utama: machine learning techniques, server-side techniques, client-side techniques, proxy-based techniques, dan combined approaches. Hasil kajian menunjukkan bahwa sebagian besar teknik pertahanan saat ini efektif untuk menangani stored XSS dan reflected XSS, namun masih belum ada solusi yang benar-benar andal untuk melindungi dari DOM-based XSS dan mutation-based XSS. Penelitian ini merekomendasikan kombinasi static analysis, dynamic analysis, code auditing, secure coding, serta kampanye kesadaran pengguna sebagai pendekatan menyeluruh dalam mitigasi XSS.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu keduanya membahas serangan XSS secara teoritis dan praktis serta pentingnya pendekatan berlapis (multi-layered security) dalam mitigasi XSS.

	Penelitian tersebut bersifat survei dan konseptual, sedangkan skripsi ini bersifat perancangan dan evaluasi arsitektur infrastruktur keamanan open source pada OJS.
No	17
Judul	*WEB VULNERABILITY SCANNER PADA JENIS SERANGAN IMPROPER PRIVILEGE MANAGEMENT AND AUTHENTICATION MENGGUNAKAN METODE TAINT ANALYSIS*
Penulis	Ratu, Maharani Sofya Laksmibai and abdurrasyid, abdurrasyid and Prayitno, Budi
Tahun	2024
Hasil	Penelitian ini mengembangkan sistem Web Vulnerability Scanner untuk mendeteksi serangan *Improper Privilege Management* dan *Improper Authentication* menggunakan metode Taint Analysis. Analisis dilakukan secara statis terhadap kode sumber aplikasi berbasis PHP tanpa menjalankan aplikasinya. Sistem diuji pada beberapa kode sumber dan hasilnya menunjukkan tingkat akurasi sebesar 99% dan recall 100%, tanpa menghasilkan kesalahan positif maupun negatif. Hal ini menunjukkan bahwa sistem sangat efektif dalam mendeteksi kerentanan keamanan aplikasi web sejak tahap awal pengembangan.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu relevan dengan skripsi karena sama-sama membahas deteksi dan mitigasi kerentanan keamanan pada aplikasi web.

	Metode Taint Analysis yang digunakan mendukung evaluasi arsitektur keamanan dalam mendeteksi potensi celah keamanan secara sistematis. Penelitian ini berfokus pada deteksi kerentanan terkait manajemen hak akses dan autentikasi sedangkan skripsi ini secara spesifik membahas mitigasi serangan Cross-Site Scripting (XSS) pada platform Open Journal Systems (OJS) dengan pendekatan arsitektur infrastruktur keamanan berbasis open source.
No	18
Judul	*CONTENT MANAGEMENT SYSTEM SECURITY RECOMMENDATIONS WITH PENETRATION TEST USING THE OWASP WEB SECURITY TEST GUIDE METHODOLOGY*
Penulis	Fadhlurrohman, Muhammad Dzaki and Sudirman, M. Yoga Distra and Putra, Rakhmadi Irfansyah
Tahun	2024
Hasil	Pengujian keamanan dilakukan pada website berbasis WordPress menggunakan metodologi OWASP Web Security Testing Guide (WSTG) dengan 11 tahapan pengujian, yaitu mulai dari Information Gathering hingga Client-side Testing. Hasil penelitian menemukan tujuh kerentanan, termasuk kebocoran informasi sensitif yang berpotensi dieksploitasi oleh penyerang. Seluruh kerentanan diklasifikasikan berdasarkan OWASP Top 10 tahun 2021. Rekomendasi perbaikan mencakup pembaruan sistem secara berkala serta peningkatan konfigurasi keamanan untuk menurunkan risiko eksploitasi.

Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu relevan dengan skripsi karena sama-sama menggunakan pendekatan pengujian keamanan berbasis standar OWASP untuk mengidentifikasi kerentanan pada aplikasi web. Metodologi OWASP WSTG yang digunakan dapat diterapkan dalam perancangan dan evaluasi arsitektur keamanan OJS. Penelitian ini berfokus pada platform WordPress sebagai CMS umum sedangkan skripsi ini berfokus pada platform Open Journal Systems (OJS) serta pada perancangan dan evaluasi arsitektur infrastruktur keamanan berbasis open source untuk mitigasi serangan Cross-Site Scripting (XSS) secara spesifik.
No	19
Judul	*Development of Journal Systems and the Use of Cosine Similarity and SVM Text Mining for Plagiarism Assessment on Titles and Abstracts*
Penulis	Mandana, Rizal Adi and Sudirman, M. Yoga Distra and Kusuma, Dine Tiara
Tahun	2018
Hasil	Penelitian ini mengembangkan fitur analisis plagiarisme pada sistem pengelolaan jurnal untuk membantu tim editorial dalam menilai keaslian artikel. Metode yang digunakan adalah SDLC Waterfall dengan penerapan Text Mining, Cosine Similarity, dan Support Vector Machine (SVM) untuk mengukur tingkat kemiripan judul dan abstrak jurnal. Pengujian dilakukan pada portal e-journal STT-PLN dengan 30 data jurnal dan lebih dari 1800 data latih. Hasil menunjukkan sistem mampu

	mendeteksi plagiarisme secara dini dengan nilai presisi 100% dan recall 100%, dengan rentang kemiripan 5,52% – 29,03%. Namun, peningkatan jumlah data latih berdampak pada waktu proses yang lebih lama.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu relevan dengan skripsi karena sama-sama membahas pengembangan dan evaluasi sistem pada platform jurnal elektronik (OJS/e-journal) untuk meningkatkan kualitas dan keamanan pengelolaan jurnal. Keduanya menekankan pentingnya fitur pendukung dalam menjaga integritas sistem jurnal. Penelitian ini berfokus pada deteksi plagiarisme menggunakan metode Text Mining, Cosine Similarity, dan SVM, sedangkan skripsi ini berfokus pada perancangan dan evaluasi arsitektur infrastruktur keamanan berbasis open source untuk mitigasi serangan Cross-Site Scripting (XSS) pada platform Open Journal Systems (OJS).
No	20
Judul	*APPLICATION OF FOOTPRINTING AND VULNERABILITY SCANNING IN IDENTIFYING WEBSITE SECURITY VULNERABILITIES CASE STUDY: PT INTERCLOUD DIGITAL INOVASI*
Penulis	Antony, Billy and Fitriani, Yessy and Sudirman, M. Yoga Distra
Tahun	2024
Hasil	Penelitian ini menganalisis kerentanan keamanan pada website PT Intercloud Digital Inovasi menggunakan metode Footprinting dan Vulnerability Scanning.

	Footprinting digunakan untuk mengumpulkan informasi mendalam mengenai target sedangkan vulnerability scanning digunakan untuk memetakan dan mengevaluasi kelemahan yang ada. Hasil penelitian menunjukkan bahwa meskipun sistem keamanan perusahaan sudah cukup baik, masih terdapat celah yang dapat dieksploitasi, seperti potensi serangan SQL Injection, Brute Force, dan DDoS. Oleh karena itu diperlukan upaya yang lebih intensif dan sistematis untuk mengidentifikasi serta memitigasi ancaman keamanan siber.
Keterkaitan Penelitian	Adapun keterkaitan penelitian di atas yaitu relevan dengan skripsi karena sama-sama berfokus pada analisis kerentanan keamanan website dan penggunaan metode pengujian keamanan untuk mengidentifikasi potensi serangan. Pendekatan footprinting dan vulnerability scanning mendukung proses evaluasi arsitektur keamanan yang juga dilakukan dalam skripsi ini. Penelitian ini tidak secara spesifik membahas mitigasi serangan Cross-Site Scripting (XSS) maupun platform Open Journal Systems (OJS) sedangkan skripsi ini berfokus pada perancangan dan evaluasi arsitektur infrastruktur keamanan berbasis open source untuk mitigasi serangan XSS pada OJS.

Berdasarkan tinjauan terhadap penelitian relevan yang telah diuraikan pada tabel di atas dapat ditarik sebuah pemahaman komprehensif mengenai tren dan tantangan dalam mengamankan aplikasi berbasis web khususnya pada platform *Open Journal Systems* (OJS). Secara garis besar fokus penelitian terdahulu dapat dipetakan ke dalam tiga arah utama yaitu identifikasi kerentanan pada OJS, pemanfaatan infrastruktur virtualisasi, serta teknik mitigasi spesifik terhadap serangan injeksi seperti *Cross-Site Scripting* (XSS).

Pada aspek identifikasi kerentanan penelitian yang dilakukan oleh (Guntoro et al., 2020), (Riadi et al., 2020), serta (Yunanri.W, 2023) secara konsisten membuktikan bahwa OJS merupakan target yang rentan terhadap berbagai eksploitasi eksternal. Pengujian keamanan menggunakan pendekatan *Vulnerability Assessment* dan *Penetration Testing* mengungkap bahwa versi-versi OJS lama memiliki celah keamanan berisiko tinggi termasuk kerentanan XSS dan SQL *Injection*. Namun mayoritas penelitian ini masih berfokus pada pelaporan kerentanan dalam arsitektur server monolitik tradisional tanpa mengusulkan atau menguji solusi mitigasi pada level infrastruktur yang menopang aplikasi tersebut. Di sisi lain beberapa kajian literatur dan survei komprehensif seperti yang dilakukan oleh (*1-10 Threat Detection*, n.d.) dan (Weamie, 2022a) menegaskan bahwa XSS tetap menjadi ancaman dominan yang membutuhkan pendekatan pertahanan berlapis mengingat teknik pertahanan tunggal seringkali gagal menangani variasi serangan XSS secara menyeluruh.

Selanjutnya terkait pada aspek infrastruktur penelitian terkait virtualisasi dan segmentasi jaringan menunjukkan potensi besar dalam meningkatkan postur keamanan. Studi oleh (Alsaqour et al., 2021), (Emmer P. Ruaya et al., 2022), serta (Pu et al., 2020) menyoroti pentingnya isolasi lalu lintas jaringan menggunakan *firewall*, VLAN, maupun teknologi VXLAN untuk mencegah penyebaran ancaman secara lateral antar sistem. Lebih spesifik lagi penelitian oleh (Koratagere et al., 2023) dan (Homepage et al., 2025) membuktikan bahwa penggunaan platform virtualisasi *open source* seperti Proxmox VE tidak hanya mengoptimalkan efisiensi sumber daya tetapi juga menyediakan fitur keamanan terintegrasi seperti isolasi *Virtual Machine* (VM) dan kontrol akses berbasis peran (RBAC). Meskipun demikian pemanfaatan virtualisasi tingkat lanjut ini umumnya masih ditujukan untuk efisiensi operasional pusat data pendidikan tinggi, bukan secara eksplisit dirancang sebagai arsitektur keamanan aktif untuk melindungi layanan publikasi jurnal dari serangan aplikasi.

Sementara itu dalam konteks mitigasi ancaman aplikasi penelitian oleh (Fadlil et al., 2022a) serta (Babaey & Ravindran, 2025) menunjukkan pergeseran paradigma pertahanan ke arah penerapan *secure coding* dan penggunaan *Web Application Firewall* (WAF) seperti ModSecurity. Pendekatan ini terbukti efektif dalam memblokir *payload* XSS yang berbahaya sebelum mencapai logika inti aplikasi. Akan tetapi implementasi

WAF seringkali diterapkan secara terpisah dan belum sepenuhnya terintegrasi dengan strategi segmentasi infrastruktur yang kokoh.

Berdasarkan *research gap* dari penelitian-penelitian terdahulu skripsi ini mengambil posisi yang jelas dan memberikan nilai kebaruan (*novelty*). Jika penelitian sebelumnya umumnya hanya berfokus pada pendeteksian celah OJS atau sekadar membangun infrastruktur virtual untuk manajemen sumber daya maka penelitian ini mengintegrasikan kedua ranah tersebut. Kebaruan penelitian ini terletak pada perancangan dan evaluasi arsitektur infrastruktur keamanan berbasis *open source* yang menggabungkan isolasi *container* berbasis Proxmox VE, pelindungan aplikasi melalui LWAF, dan keamanan berlapis secara spesifik untuk memitigasi serangan XSS pada platform OJS. Dengan menguji arsitektur ini pada berbagai variasi kombinasi arsitektur seperti Web Server (Apache/Nginx), Database (MariaDB/PostgreSQL), PHP Version (7.4/8.0), Apps Version (3.3.0-16/3.4.0-5) penelitian ini diharapkan mampu menghasilkan rekomendasi konfigurasi pertahanan yang paling optimal dan komprehensif untuk digunakan bagi institusi pengelola jurnal ilmiah.

2.2 Landasan Teori

Sebagai landasan melakukan penelitian maka dibahas berbagai konsep dan teori yang digunakan dalam penelitian ini yang berkaitan. Berikut ini diuraikan landasan teori dari penelitian ini.

2.2.1 Keamanan Sistem Informasi

A. Keamanan Sistem Informasi

Keamanan sistem informasi merupakan metode untuk melindungi sistem informasi dari ancaman seperti akses yang tidak sah, gangguan, pencurian, atau kerusakan. Ini mencakup perlindungan terhadap perangkat keras, perangkat lunak, data, jaringan, dan pengguna sistem terutama pada sistem yang menangani data sensitif seperti sistem publikasi ilmiah berbasis *Open Journal System* (OJS). Menurut Whitman & Mattord, 2018) mendefinisikan bahwa keamanan informasi sebagai upaya untuk melindungi sistem informasi termasuk perangkat keras, perangkat lunak, data, dan jaringan tetapi tidak hanya itu keamanan informasi tidak hanya berkaitan dengan perlindungan terhadap perangkat keras atau perangkat lunak melainkan juga pada jaminan terhadap tiga prinsip utama atau pilar utama yang dikenal sebagai *CIA Triad* yang

terdiri dari *kerahasiaan* (confidentiality), *integritas* (integrity), dan *ketersediaan* (availability).

Menurut (Stallings et al., 2012) mendifinisakan bahwa keamanan sistem informasi tidak hanya terbatas pada aspek teknis tetapi juga mencakup kebijakan, prosedur, serta kontrol administratif yang memastikan bahwa informasi hanya dapat diakses oleh pihak yang berwenang tetap utuh tanpa perubahan yang tidak sahcdan tersedia ketika dibutuhkan. Dengan demikian keamanan sistem informasi dapat dipandang sebagai kombinasi antara kontrol teknologi contohnya enkripsi, firewall, otentikasi, kontrol administratif contohnya SOP kebijakan akses serta kontrol fisik contohnya pengamanan server di ruang khusus.

Dalam konteks penelitian ini keamanan sistem informasi menjadi landasan utama karena sistem *Open Journal Systems* (OJS) menyimpan informasi penting berupa akun pengguna, metadata artikel ilmiah, serta dokumen penelitian. Tanpa penerapan keamanan yang kurang memadai informasi-informasi terkait diatas tadi berisiko mengalami kebocoran, perubahan data, hingga hilangnya akses layanan.

B. CIA Triad (Confindentiality, Integrity, Availability)

Model dasar keamanan informormasi yang berfokus pada 3 pilar utama yang terdiri dari:

1. Confindentiality (Kerahasiaan)

   Prinsip ini memastikan bahwa informasi sensitif hanya dapat diakses oleh individu atau entitas yang berwenang. Untuk mencapai kerahasiaan suatu organisasi menerapkan kebijakan dan teknologi seperti enkripsi, kontrol akses berbasis peran, dan otentikasi multifaktor. Contohnya dalam penelitian ini dalam arsitektur OJS confidentiality dijaga dengan menempatkan *Database* dan *Data Files* terpisah dari *Aplikasi*. Hal ini memastikan bahwa meskipun aplikasi web OJS terkena serangan contohnya seperti brute force login atau eksploitasi plugin, data sensitif seperti akun pengguna dan file artikel tetap memiliki lapisan proteksi tambahan keamanan.

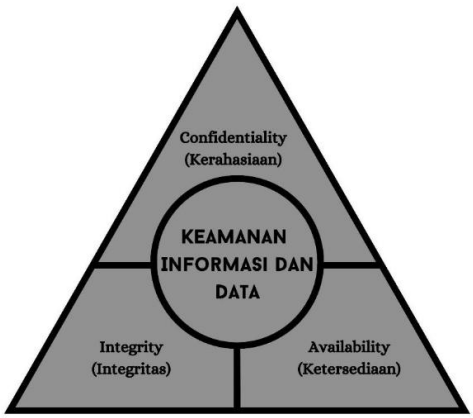
2. Integrity (Integritas)

   Prinsip integritas menjamin keotentikan dan keakuratan data memastikan bahwa informasi tidak diubah atau dimanipulasi secara tidak sah. Contohnya

dalam penelitian ini integritas dijaga melalui pemisahan fungsi database. Dengan adanya *Database* tersendiri ancaman seperti SQL Injection dapat lebih mudah diisolasi sehingga tidak langsung merusak data inti jurnal.

3. Availability (Ketersediaan)

   Prinsip ini memastikan bahwa sistem, data, dan layanan dapat diakses oleh pengguna yang berwenang saat dibutuhkan. Hal ini dicapai melalui strategi seperti pencadangan data, perencanaan pemulihan bencana, dan manajemen kapasitas. Gangguan seperti serangan Denial of Service (DoS) dapat memengaruhi ketersediaan data oleh karena itu persiapan yang matang sangat penting. Contohnya dalam penelitian ini Dalam konteks Open Journal System (OJS) availability dijaga melalui penerapan virtualisasi Proxmox yang memudahkan proses pemeliharaan, migrasi, maupun pemulihan ketika terjadi gangguan. Dengan dilakukannya pemisahan antara App, DB, dan Data Files kerusakan pada salah satu komponen tidak langsung menyebabkan seluruh sistem tidak tersedia.



Gambar 2. 1 CIA TRIAD (Harahap et al., n.d.)

Dengan demikian dari penjelasan diatas CIA Triad menjadi fondasi utama dalam merancang arsitektur keamanan Open Journal Sytem (OJS) pada penelitian ini. Setiap lapisan arsitektur yang diterapkan diarahkan untuk mendukung tercapainya confidentiality, integrity, dan availability sesuai standar keamanan informasi (*27001-2013-Technical-Guidance*, n.d.)

2.2.2 Virtualisasi dan Proxmox Virtual Environment (VE)

Virtualisasi merupakan teknologi abstraksi perangkat keras yang memungkinkan satu fisik komputer (*host*) untuk menjalankan beberapa sistem operasi atau lingkungan

komputasi yang terisolasi satu sama lain secara bersamaan. Dalam arsitektur keamanan modern virtualisasi berperan penting sebagai lapisan pertahanan pertama melalui mekanisme isolasi sumber daya. Hal ini memastikan bahwa apabila terjadi kegagalan atau kompromi keamanan pada satu layanan dampaknya tidak akan menyebar secara langsung ke layanan lainnya dalam infrastruktur yang sama (Koratagere et al., 2023).

Salah satu platform virtualisasi berbasis *open source* tingkat perusahaan yang paling andal adalah Proxmox Virtual Environment (VE). Proxmox VE merupakan solusi manajemen virtualisasi lengkap yang menggabungkan dua teknologi utama yaitu *Kernel-based Virtual Machine* (KVM) untuk virtualisasi penuh (*full virtualization*) dan *Linux Containers* (LXC) untuk virtualisasi tingkat sistem operasi yang ringan. Proxmox dipilih dalam perancangan arsitektur keamanan OJS karena menyediakan fitur keamanan terintegrasi seperti *firewall* pada level *hypervisor*, *Role-Based Access Control* (RBAC), serta dukungan terhadap sistem penyimpanan data yang aman dan terdistribusi (J, Billpurta A et al., 2025).

Penerapan Proxmox VE dalam penelitian ini bertujuan untuk mengimplementasikan konsep *segregation of duties* dan mitigasi serangan lateral. Dengan memisahkan server aplikasi OJS dan basis data ke dalam *container* atau VM yang berbeda peneliti dapat membatasi ruang gerak penyerang. Jika penyerang berhasil melakukan eksploitasi XSS pada lapisan aplikasi teknologi isolasi pada Proxmox akan mempersulit penyerang untuk menembus lapisan basis data atau melakukan *privilege escalation* ke tingkat sistem operasi fisik (Pu et al., 2020). Selain itu fitur *snapshot* pada Proxmox memungkinkan peneliti untuk menjamin *reproducibility* pengujian dengan cara mengembalikan kondisi server ke keadaan semula sebelum serangan dilakukan (Prabowo et al., 2024).

2.2.3 Cross Site Scripting

Cross-Site Scripting (XSS) merupakan salah satu kerentanan keamanan aplikasi web paling umum dan berbahaya, di mana penyerang menyisipkan skrip berbahaya umumnya JavaScript ke dalam halaman web yang kemudian dieksekusi di browser korban. Serangan ini memanfaatkan kelemahan validasi input pada aplikasi web, sehingga memungkinkan eksekusi kode pihak ketiga dalam konteks sesi pengguna yang sah (Saputra & Hidayat, 2025).

Menurut (OWASP Top 10, 2021) XSS menempati posisi kritis dalam daftar OWASP Top 10 sebagai bagian dari kategori Injection, dan tetap menjadi ancaman utama terhadap platform berbasis web termasuk sistem manajemen jurnal akademik seperti *Open Journal Systems* (OJS). XSS tidak hanya mengancam integritas konten tetapi juga dapat mencuri sesi pengguna, mengalihkan alur navigasi, atau bahkan menginstal backdoor pada sistem (Saputra & Hidayat, 2025); (Riadi et al., 2020).

Secara umum XSS diklasifikasikan ke dalam tiga jenis utama:

A. Stored XSS

Terjadi ketika payload skrip berbahaya disimpan secara permanen di server misalnya dalam database dan dieksekusi setiap kali pengguna mengakses halaman yang memuat konten tersebut. Pada OJS, Stored XSS sering ditemukan pada fitur diskusi, komentar, atau metadata artikel yang memungkinkan input HTML tanpa sanitasi yang memadai (Saputra & Hidayat, 2025).

B. Reflected XSS

Payload XSS dikirim melalui parameter URL (misalnya melalui parameter pencarian atau error message) dan langsung dieksekusi di browser korban. Serangan ini biasanya disebar melalui tautan jahat yang dikirim melalui email atau pesan instan (Riadi et al., 2020).

C. DOM-based XSS

Eksploitasi terjadi sepenuhnya di sisi klien melalui manipulasi Document Object Model (DOM) oleh skrip yang tidak aman. Berbeda dengan dua jenis sebelumnya, DOM-based XSS tidak melibatkan server dalam eksekusi payload (Yunanri.W, 2023).

Dalam konteks OJS penelitian oleh (Saputra & Hidayat, 2025) berhasil mengeksploitasi kerentanan Stored XSS melalui fitur discussion panel pada OJS versi 3.1.1-4. Penyerang dapat menyisipkan skrip berbahaya dalam judul diskusi yang aktif saat artikel memasuki tahap review. Ketika pengelola jurnal mengakses diskusi tersebut, skrip tersebut secara otomatis mengeksekusi fungsi pencurian sesi (session hijacking), memungkinkan penyerang mengakses sistem dengan hak akses administrator tanpa perlu autentikasi ulang.

Temuan serupa juga dilaporkan oleh (Riadi et al., 2020) yang mengidentifikasi XSS sebagai salah satu dari 6.049 kerentanan pada OJS versi 2.4.7. Pengujian menggunakan OWASP ZAP menunjukkan bahwa banyak komponen OJS tidak menerapkan sanitasi input atau *output encoding* yang memadai, sehingga rentan terhadap injeksi skrip.

Serangan XSS pada OJS sangat berbahaya karena:

A. Pengelola jurnal dan editor sering memiliki hak akses tinggi (manajemen pengguna, konfigurasi sistem, unggah file).
B. OJS memungkinkan pengguna memasukkan konten HTML (misalnya pada abstrak, metadata, atau komentar), yang menjadi vektor ideal untuk injeksi XSS.
C. Banyak institusi masih menggunakan versi OJS lama yang belum diperbarui, sehingga kerentanan tetap terbuka ((Riadi et al., 2020)(Saputra & Hidayat, 2025).

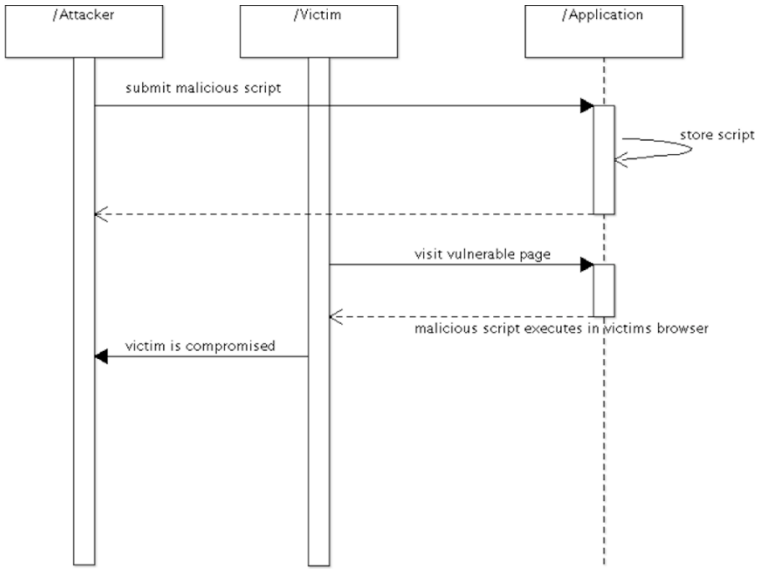
Untuk mencegah XSS pendekatan mitigasi harus bersifat berlapis:

A. Sanitasi input: Membersihkan atau menolak input yang mengandung karakter HTML/JavaScript berbahaya.
B. Output encoding: Mengonversi karakter khusus (seperti <, >, &) menjadi entitas HTML sebelum ditampilkan di browser.
C. Content Security Policy (CSP): Membatasi sumber eksternal yang boleh mengeksekusi skrip di halaman web.
D. Penggunaan atribut keamanan pada cookie: Seperti HttpOnly dan SameSite untuk mencegah pencurian sesi melalui XSS (Yunanri.W, 2023).

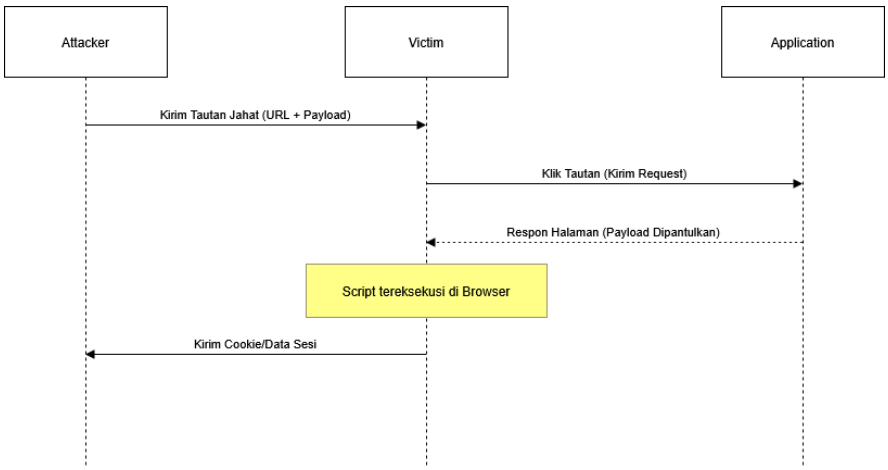
Dalam penelitian ini fokus utama adalah pada identifikasi dan mitigasi XSS pada OJS melalui desain arsitektur infrastruktur keamanan yang mengintegrasikan lapisan pertahanan teknis seperti header keamanan dan isolasi komponen bukan hanya perbaikan pada kode aplikasi. Pendekatan ini selaras dengan prinsip *defense in depth* di mana keamanan tidak hanya bergantung pada lapisan aplikasi, tetapi juga pada konfigurasi infrastruktur yang tangguh.

Untuk memahami alur kerja serangan XSS secara mendalam khususnya pada jenis *Stored XSS* yang sering menargetkan platform manajemen jurnal maka mekanisme interaksi antara penyerang (*attacker*), aplikasi (*application*), dan korban (*victim*) dapat

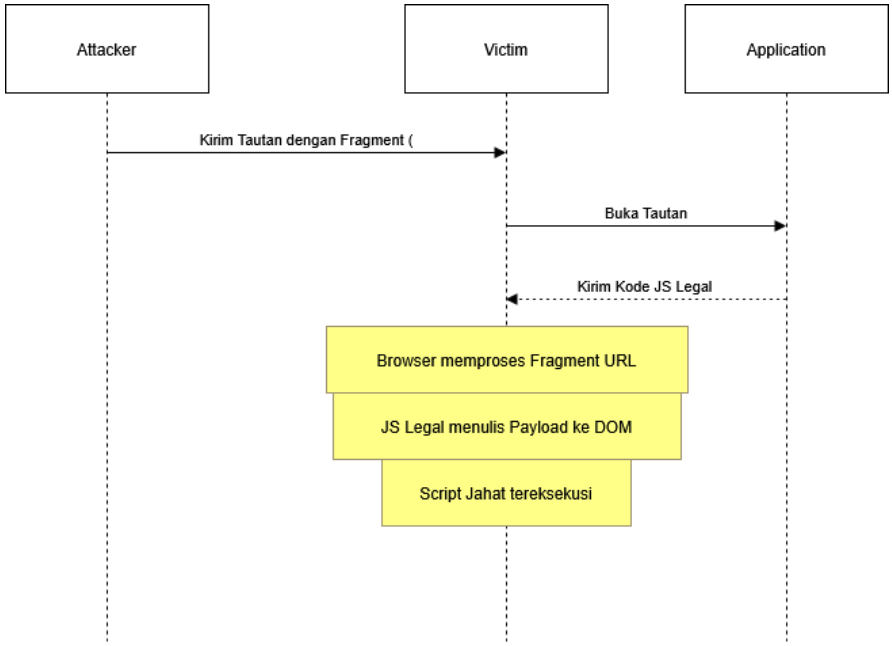
digambarkan melalui sebuah diagram urutan (*sequence diagram*) pada Gambar 2.5, 2.6, 2.7 berikut:



Gambar 2. 2 Alur Serangan Stored XSS



Gambar 2. 3 Alur Serangan Reflected XSS



Gambar 2. 4 Alur Serangan DOM-Based XSS

Mekanisme serangan Cross-Site Scripting (XSS) pada platform *Open Journal Systems* (OJS) dapat diklasifikasikan menjadi tiga jenis utama berdasarkan cara penyampaian skrip jahat dan lokasi eksekusinya sebagaimana digambarkan dalam rangkaian sequence diagram diatas berikut.

Pertama, jenis serangan yang paling berbahaya adalah Stored XSS yang diilustrasikan pada Gambar 2.2. Alur serangan ini bersifat permanen karena payload atau skrip jahat dikirimkan oleh penyerang (attacker) melalui fitur input aplikasi yang kemudian disimpan langsung ke dalam basis data server. Ketika pengguna lain atau administrator (victim) mengunjungi halaman yang memuat data terinfeksi tersebut server secara otomatis mengirimkan skrip jahat ke browser korban. Akibatnya skrip tereksekusi di sisi klien yang berujung pada pengambilalihan sesi atau pencurian data sensitif tanpa disadari oleh korban.

Kedua, berbeda dengan sifatnya yang permanen Reflected XSS pada Gambar 2.3 mengandalkan teknik pantulan melalui parameter HTTP. Alur dimulai dengan penyerang mengirimkan tautan jahat yang telah disisipi payload kepada korban melalui teknik social engineering. Saat korban mengklik tautan tersebut browser akan mengirimkan permintaan (request) ke aplikasi OJS. Aplikasi kemudian menerima input tersebut dan langsung memantulkannya kembali dalam halaman respon tanpa melalui proses

penyimpanan di database. Browser korban yang menganggap respon tersebut berasal dari sumber terpercaya akan mengeksekusi skrip jahat tersebut dan secara otomatis mengirimkan data sesi atau cookie kembali ke pihak penyerang.

Ketiga, DOM-based XSS yang ditunjukkan pada Gambar 2.4 memiliki karakteristik unik di mana serangan terjadi sepenuhnya pada sisi klien tanpa melibatkan server aplikasi dalam pengolahan payload. Penyerang mengirimkan tautan yang mengandung skrip jahat pada bagian fragment URL (setelah tanda #). Ketika korban membuka tautan tersebut aplikasi OJS hanya mengirimkan kode JavaScript legal yang bertugas mengatur tampilan halaman. Namun karena adanya celah pada skrip JavaScript di sisi klien browser memproses fragment URL tersebut secara dinamis dan menuliskan payload jahat langsung ke dalam struktur *Document Object Model* (DOM) halaman web. Hal ini menyebabkan skrip jahat tereksekusi di dalam browser korban meskipun server aplikasi tidak pernah menerima atau memproses payload tersebut.

Secara keseluruhan meskipun ketiga jenis XSS ini memiliki jalur eksploitasi yang berbeda dampak akhir yang ditimbulkan memiliki kesamaan yaitu pelanggaran terhadap pilar kerahasiaan (confidentiality) dan integritas (integrity) data melalui eksekusi kode pihak ketiga yang tidak sah dalam konteks sesi pengguna yang valid.

2.2.4 Karakteristik Teknologi Infrastruktur (Web Server & DBMS)

Infrastruktur pendukung aplikasi merupakan komponen vital dalam menentukan performa dan ketahanan sebuah sistem informasi. Dalam perancangan arsitektur keamanan OJS pemilihan *Web Server* dan DBMS menjadi variabel yang diuji untuk mengevaluasi bagaimana setiap teknologi berinteraksi dengan lapisan mitigasi seperti LWAF dan CSP.

A. Web Server: Apache HTTP Server vs Nginx

Web Server bertanggung jawab untuk menerima permintaan HTTP dari klien dan mengirimkan kembali respon berupa konten web. Dua teknologi yang paling dominan digunakan adalah Apache dan Nginx, yang memiliki karakteristik berbeda dari sisi arsitektur dan penanganan keamanan:

1. Apache HTTP Server

   Menggunakan arsitektur berbasis proses di mana setiap permintaan dapat ditangani oleh satu *thread* atau proses terpisah. Keunggulan utama Apache dalam

penelitian ini memungkinkan integrasi WAF yang sangat stabil dan kaya fitur (Arta et al., n.d.).

2. Nginx

Menggunakan arsitektur berbasis kejadian yang dirancang untuk menangani ribuan koneksi secara bersamaan dengan penggunaan sumber daya yang sangat rendah. Nginx sering digunakan sebagai *Reverse Proxy* di depan aplikasi web untuk meningkatkan keamanan melalui isolasi trafik. Namun penelitian menunjukkan bahwa pada kondisi beban tinggi Nginx memiliki *throughput* yang cepat tetapi memerlukan konfigurasi keamanan yang lebih teliti dibandingkan Apache untuk menjaga stabilitas sistem jurnal (Arta et al., n.d.).

B. Database Management System (DBMS): MariaDB vs PostgreSQL

Sistem manajemen basis data merupakan tempat penyimpanan seluruh informasi krusial jurnal mulai dari akun pengguna hingga naskah ilmiah. Keamanan pada level basis data sangat penting untuk memitigasi dampak akhir dari serangan injeksi seperti XSS (Rizal & Mandana, n.d.)

1. MariaDB

Merupakan pengembangan dari MySQL yang bersifat *open source* dan merupakan basis data standar yang didukung secara luas oleh platform OJS. MariaDB dikenal karena kemudahannya dalam konfigurasi dan kinerjanya yang optimal untuk operasi baca-tulis (*read-heavy*) pada sistem informasi publikasi jurnal (J, Billpurta A et al., 2025).

2. PostgreSQL

Merupakan sistem basis data objek-relasional yang lebih maju yang menekankan pada standar integritas data dan keamanan yang lebih ketat. PostgreSQL menyediakan fitur keamanan tingkat lanjut yang lebih kompleks dibandingkan MariaDB sehingga seringkali dipilih untuk infrastruktur yang membutuhkan tingkat keandalan data yang sangat tinggi (Saputra & Hidayat, 2025).

Penyediaan berbagai variasi infrastruktur ini bertujuan untuk membuktikan bahwa arsitektur keamanan yang dirancang LWAF dan CSP bersifat agnostik dan

efektif terlepas dari kombinasi *web server* atau basis data yang digunakan oleh institusi pengelola jurnal.

2.2.5 Open Journal System (OJS)

Open Journal Systems (OJS) merupakan perangkat lunak berbasis web yang dikembangkan oleh *Public Knowledge Project (PKP)* dan digunakan secara luas oleh institusi akademik untuk mengelola proses penerbitan jurnal ilmiah secara daring. Karena sifatnya yang *open source* dan diakses melalui internet OJS berpotensi menjadi target berbagai serangan siber baik pada lapisan aplikasi maupun konfigurasi server.

Menurut (Riadi et al., 2020) OJS adalah perangkat lunak publikasi ilmiah yang digunakan secara global. OJS rentan diserang oleh peretas yang dapat menurunkan kinerja dan keamanannya. Permasalahan yang sering muncul meliputi jaringan, port discovery, dan audit eksploitasi sistem seperti *SQL Injection* serta pembobolan kata sandi. Penelitian ini menggunakan metode *vulnerability assessment* dengan tahapan *information gathering*, *vulnerability scanning*, dan *reporting*, serta mengacu pada standar *OWASP*. Hasil pengujian menemukan 6.049 kerentanan, terdiri dari 70 *high*, 1.929 *medium*, dan 4.050 *low* OJS yang digunakan versi 2.4.7 memiliki banyak celah keamanan dan tidak direkomendasikan untuk digunakan, sehingga disarankan memakai versi terbaru dari *Public Knowledge Project (PKP)*.

Selain *SQL injection* salah satu ancaman yang sering ditemukan dalam OJS adalah *Cross-Site Scripting* (XSS). Studi lain dari (Saputra & Hidayat, 2025) Sistem *Open Journal Systems* (OJS) yang dikembangkan oleh *Public Knowledge Project* memiliki beberapa versi salah satunya versi 3.1.1.4 yang masih banyak digunakan oleh berbagai institusi sebagai platform publikasi karya ilmiah tetapi versi tersebut memiliki kerentanan keamanan salah satunya berupa *Cross-Site Scripting* (XSS). Kerentanan ini dapat dimanfaatkan oleh pelaku kejahatan siber untuk mengambil alih sesi manajer jurnal dan melakukan tindakan ilegal seperti menanamkan *backdoor* pada sistem dan juga penyerang dapat menyisipkan skrip berbahaya dalam diskusi untuk mencuri sesi.

Selain kerentanan pada injeksi ada juga salah satu kerentanan bernama *misconfinguration* juga menjadi salah satu permasalahan dalam OJS. Temuan ini memperkuat rekomendasi dari salah satu pengembang inti dari PKP yaitu Alex Smecher

yang menyarankan agar direktori *files_dir* dipindahkan ke luar direktori publik server web serta hak aksesnya dibatasi hanya untuk proses yang dijalankan oleh aplikasi OJS.

2.2.6 Defense In Depth

Defense in Depth merupakan salah satu strategi atau konsep dalam keamanan yang menekankan penerapan berlapis yang bertujuan untuk melindungi aset, data, dan sistem komputer dari ancaman serangan. Strategi ini sangat krusial diimplementasikan pada aplikasi publik seperti *Open Journal Systems* (OJS) karena karakteristik platform tersebut yang menangani data ilmiah sensitif dan sering menjadi target eksploitasi (Riadi et al., 2020).

Dalam konteks keamanan Open Journal System (OJS) *defense in depth* menjadi sangat krusial karena OJS merupakan aplikasi web publik yang:

A. Menangani data sensitif (naskah ilmiah, identitas penulis/reviewer),
B. Memiliki titik masuk yang luas (form submission, diskusi, komentar),
C. Sering menggunakan versi lama yang rentan terhadap eksploitasi (Riadi et al., 2020)

Berdasarkan analisis terhadap penelitian terdahulu (Guntoro et al., 2020) arsitektur keamanan OJS dapat dikembangkan dalam empat lapisan utama:

1. Lapisan Perimeter (Perimeter Defense)

Lapisan ini merupakan garis pertahanan pertama terhadap ancaman eksternal. Pada penelitian ini, lapisan perimeter diwujudkan melalui RouterOS sebagai firewall utama yang ditempatkan di batas antara internet dan jaringan internal. RouterOS dikonfigurasi untuk:

A. Melakukan *packet filtering* berdasarkan alamat IP dan port,
B. Menerapkan *rate limiting* untuk mitigasi serangan Denial of Service (DoS),
C. Mengaktifkan *Layer 7 filtering* untuk memblokir pola lalu lintas berbahaya

Studi oleh (Guntoro et al., 2020) menunjukkan bahwa OJS Universitas Lancang Kuning rentan terhadap serangan DoS, namun tidak terhadap SQL Injection. Hal ini memperkuat kebutuhan akan firewall yang mampu mendeteksi dan membatasi lalu lintas anomali sejak di lapisan perimeter.

2. Lapisan Jaringan Internal (Internal Network Defense)

Setelah melewati perimeter, lalu lintas yang sah memasuki jaringan internal yang telah disegmentasi secara logis. Dalam penelitian ini, segmentasi dicapai melalui virtualisasi berbasis Proxmox VE yang memungkinkan pemisahan fungsi ke dalam mesin virtual (VM) terpisah:

A. VM Aplikasi: menjalankan OJS dan web server,
B. VM Database: menyimpan data jurnal,
C. VM Data Files: menyimpan file artikel di luar *web root*.

Pendekatan ini selaras dengan prinsip *segregation of duties* dan *least privilege*. Seperti yang ditekankan oleh (Prabowo et al., 2024) kesalahan konfigurasi pada lingkungan dapat memberikan penyerang kendali penuh atas sistem. Oleh karena itu, isolasi VM melalui Proxmox bukan hanya untuk efisiensi, tetapi juga sebagai mekanisme keamanan aktif.

3. Lapisan Aplikasi (Application Defense)

Lapisan ini melindungi kode dan logika bisnis OJS dari eksploitasi langsung. Ancaman utama di lapisan ini adalah Cross-Site Scripting (XSS), yang telah terbukti berhasil dieksploitasi melalui fitur diskusi pada OJS versi 3.1.1-4 (Saputra & Hidayat, 2025). Untuk memitigasi risiko ini, pertahanan di lapisan aplikasi mencakup:

A. Sanitasi input pada semua form (termasuk judul diskusi),
B. Penerapan header keamanan HTTP seperti *Content-Security-Policy* (CSP), *X-XSS-Protection*, dan *X-Content-Type-Options* (Yunanri, 2023),
C. Pembatasan hak akses pengguna sesuai peran (misalnya, reviewer tidak bisa mengunggah file eksekusi).

Temuan (Riadi et al., 2020) yang mengidentifikasi 70 kerentanan high-risk pada OJS 2.4.7 termasuk XSS dan header keamanan yang hilang menegaskan urgensi penguatan di lapisan ini.

4. Lapisan Data (Data Defense)

Lapisan terdalam ini melindungi integritas dan kerahasiaan data itu sendiri. Strategi utama meliputi:

A. Penyimpanan direktori files_dir di luar *web root*, sehingga file tidak dapat diakses langsung melalui URL.

B. Backup rutin menggunakan fitur snapshot Proxmox untuk pemulihan cepat pasca-serangan,
C. Enkripsi data sensitif saat disimpan atau ditransmisikan.

(Guntoro et al., 2020) mencatat bahwa OJS Universitas Lancang Kuning pernah mengalami *cracking* dua kali, yang mengakibatkan hilangnya data. Hal ini menunjukkan bahwa tanpa perlindungan di lapisan data seluruh sistem tetap rentan meskipun lapisan lain telah diamankan. Dengan pendekatan ini serangan XSS yang berhasil melewati satu lapisan tidak langsung kena seluruh sistem karena akan dihadang oleh lapisan berikutnya. Hal ini selaras dengan prinsip *defense in depth* yang direkomendasikan oleh National Institute of Standards and Technology (O'Reilly et al., 2021) pendekatan berlapis ini menjamin integritas dan ketersediaan sistem informasi meskipun menghadapi variasi serangan siber yang terus berevolusi..

2.2.7 Penetration Testing

Penetration Testing (pengujian penetrasi) merupakan pendekatan sistematis untuk mentransmisikan sistem keamanan informasi dengan menyertakan serangan nyata dari penyerang eksternal maupun internal. Metode ini bertujuan untuk mengidentifikasi, mengeksploitasi, dan menganalisis kerentanan sebelum dimanfaatkan oleh pihak jahat (Guntoro et al., 2020). Dalam konteks keamanan Open Journal Systems (OJS) *pengujian penetrasi* telah menjadi standar evaluasi dalam berbagai penelitian terdahulu karena kemampuannya mengungkap celah keamanan yang tidak terdeteksi melalui konfigurasi audit biasa.

Pada penelitian ini *penetration testing* digunakan sebagai metode utama untuk efektivitas arsitektur infrastruktur keamanan yang dirancang. Pendekatan menggunakan metode black-box di mana pengujian dilakukan tanpa akses ke kode sumber atau konfigurasi sistem internal sehingga meniru skenario serangan nyata dari luar ((Teguh Yuwono et al., 2021). Metode ini selaras dengan kerangka kerja internasional seperti OWASP Testing Guide dan ISSAF (Information Systems Security Assessment Framework) yang secara eksplisit merekomendasikan simulasi serangan sebagai bagian integral dari evaluasi keamanan aplikasi web (Guntoro et al., 2020)

Proses *pengujian penetrasi* dalam penelitian ini mencakup tiga tahap utama:

A. Information Gathering: Mengumpulkan tentang target (OJS) untuk memetakan permukaan serangan (Riadi et al., 2020)
B. Vulnerability Scanning dan Exploitation : Melakukan pemindaian terhadap kerentanan XSS menggunakan OWASP ZAP tools ini mampu mendeteksi berbagai jenis XSS (stored, Reflected, DOM-based) serta menguji serangan melalui form input, parameter URL, dan metadata ((Teguh Yuwono et al., 2021); (Riadi et al., 2020). Selanjutnya payload XSS diperkenalkan untuk menguji apakah sesi pengelola jurnal sebagaimana keberhasilan dilakukan oleh (Saputra & Hidayat, 2025) melalui fitur diskusi OJS.
C. Analisis and Reports: Hasil pengujian dibandingkan antara konfigurasi dasar OJS dan OJS yang telah diperkuat dengan arsitektur keamanan berlapis. Pengurangan jumlah payload XSS yang berhasil dieksekusi menjadi indikator utama keberhasilan mitigasi.

2.2.8 Light Weight Web Application Firewall (LWAF)

Web Application Firewall (WAF) merupakan sistem keamanan yang bekerja pada lapisan aplikasi (Layer 7) dalam model OSI yang dirancang khusus untuk memantau, menyaring, dan memblokir lalu lintas HTTP/HTTPS yang menuju ke aplikasi web. Berbeda dengan *firewall* jaringan tradisional yang fokus pada alamat IP dan *port* WAF melakukan inspeksi mendalam terhadap isi paket data (*Deep Packet Inspection*) untuk mendeteksi pola serangan spesifik aplikasi web seperti *Cross-Site Scripting* (XSS) dan SQL *Injection* (Alsaqour et al., 2021).

Salah satu implementasi WAF berbasis *open source* yang digunakan dalam penelitian ini adalah *Light Weight Web Application Firewall* (LWAF) yang diimplementasikan dalam bentuk skrip keamanan sisi server (*server-side security middleware*). Berdasarkan penelitian (Fadlil et al., 2022) penerapan WAF di depan aplikasi web mampu memberikan perlindungan secara *real-time* terhadap serangan injeksi tanpa harus melakukan perubahan pada kode sumber asli aplikasi.

Berbeda dengan WAF konvensional yang bersifat monolitik dan membutuhkan konsumsi sumber daya yang besar LWAF dirancang untuk bekerja secara spesifik dan

efisien pada lingkungan yang terisolasi seperti LXC. Mekanisme kerja LWAF terdiri dari beberapa komponen inti sebagai berikut:

1. Interseptor Paket Data: LWAF bertindak sebagai interseptor aktif yang berada di antara permintaan pengguna (*client request*) dan logika aplikasi OJS. Setiap data yang dikirimkan melalui metode POST seperti pada form registrasi wajib melewati proses inspeksi terlebih dahulu sebelum diperbolehkan untuk diproses oleh sistem basis data(Alsaqour et al., 2021).
2. Deep Packet Inspection (DPI): LWAF melakukan pemeriksaan mendalam terhadap muatan data (*payload*) pada variabel-variabel target. Proses ini bertujuan untuk mendeteksi apakah data tersebut mengandung teks murni atau instruksi pemrograman yang berbahaya(Babaey & Ravindran, 2025).
3. Filtrasi Berbasis Regular Expression (Regex): Komponen utama dalam pendeteksian serangan XSS pada LWAF adalah penggunaan teknik Regex. Regex digunakan sebagai *engine* pencocokan pola (*pattern matching*) untuk mengidentifikasi keberadaan *string* yang termasuk dalam daftar terlarang (*blacklist*) seperti tag <script>, atribut *event handler* (onerror, onload, onmouseover) hingga fungsi eksekusi JavaScript berbahaya seperti eval() dan alert().
4. Aksi Pemutusan Koneksi (Blocking): Apabila sistem mendeteksi adanya kecocokan pola antara input pengguna dengan pola serangan XSS. LWAF secara otomatis akan memutus alur eksekusi, mengirimkan respon kode status HTTP 403 Forbidden, dan mencegah data tersebut tersimpan secara permanen ke dalam database(*1-10 Threat Detection*, n.d.).

Penggunaan LWAF dalam arsitektur keamanan infrastruktur ini memberikan keunggulan berupa perlindungan yang bersifat agnostik dan ringan. Hal ini memungkinkan platform OJS memiliki lapisan pertahanan aktif tanpa harus melakukan modifikasi pada kode sumber asli aplikasi sehingga integritas sistem tetap terjaga dengan performa yang optimal.

2.2.9 Content Security Policy (CSP)

Content Security Policy (CSP) adalah lapisan keamanan tambahan yang diimplementasikan melalui *HTTP response header* untuk mendeteksi dan memitigasi

jenis serangan tertentu, termasuk *Cross-Site Scripting* (XSS) dan serangan injeksi data. CSP memungkinkan administrator sistem untuk membatasi sumber daya (seperti JavaScript, CSS, gambar) yang diizinkan untuk dimuat dan dieksekusi oleh *browser* pengguna pada platform OJS. Dengan mendefinisikan kebijakan yang ketat CSP dapat mencegah eksekusi skrip berbahaya meskipun penyerang berhasil menyisipkan *payload* ke dalam basis data atau parameter URL (Yunanri, 2023)

Mekanisme kerja CSP didasarkan pada prinsip *allowlisting*. Kebijakan CSP menginstruksikan *browser* untuk hanya mempercayai skrip yang berasal dari sumber yang telah ditentukan secara eksplisit (seperti *domain* lokal sistem jurnal). Salah satu keunggulan utama CSP dalam memitigasi XSS adalah kemampuannya untuk menonaktifkan eksekusi *inline scripts* (skrip yang ditulis langsung di dalam tag HTML) dan penggunaan fungsi-fungsi berbahaya seperti eval(). Hal ini sangat efektif untuk melindungi fitur-fitur interaktif pada OJS seperti panel diskusi dan pengelolaan naskah yang sering kali menjadi target utama serangan *Stored XSS* (Saputra & Hidayat, 2025)

Dalam arsitektur keamanan yang dirancang CSP bertindak sebagai lapisan pertahanan kedua (*secondary defense*) setelah *Web Application Firewall* (WAF). Pendekatan pertahanan berlapis ini direkomendasikan karena memberikan perlindungan yang lebih komprehensif dibandingkan teknik tunggal. Jika sebuah pola serangan XSS baru (*zero-day*) berhasil melewati filter LWAF, kebijakan CSP pada sisi klien akan tetap mencegah skrip tersebut mengirimkan data sensitif (seperti *session cookies*) ke server milik penyerang (Weamie, 2022b)

2.2.10 Metodologi Vulnerability Assessment (VA)

Vulnerability Assessment (VA) merupakan sebuah proses sistematis yang bertujuan untuk mengidentifikasi, mengukur, dan memprioritaskan kerentanan keamanan dalam sebuah sistem informasi, aplikasi, maupun infrastruktur jaringan. Dalam konteks keamanan siber metodologi VA digunakan untuk memetakan kelemahan teknis yang dapat dieksploitasi oleh penyerang sebelum ancaman tersebut benar-benar terjadi. Penekanan utama dari metodologi ini adalah pada kelengkapan identifikasi celah keamanan guna memberikan gambaran menyeluruh mengenai postur keamanan sistem.

Proses *Vulnerability Assessment* umumnya dilakukan melalui tahapan-tahapan terstruktur sebagai berikut:

1. Information Gathering (Pengumpulan Informasi): Merupakan tahap awal untuk mengumpulkan data mengenai target, mencakup identifikasi alamat IP, pemetaan struktur direktori, deteksi versi layanan (*fingerprinting*), serta pemetaan fitur-fitur aplikasi yang memiliki interaksi input dari pengguna (Untuk et al., n.d.)
2. Vulnerability Detection (Scanning): Tahap ini melibatkan penggunaan alat pemindaian otomatis (*automated scanning tools*) untuk mencari pola kerentanan yang diketahui. Dalam penelitian ini, pemindaian difokuskan pada deteksi skrip berbahaya yang berkaitan dengan kerentanan XSS (Yunanri, 2023)
3. Manual Verification: Tahap verifikasi dilakukan untuk memastikan bahwa temuan dari alat pemindaian otomatis benar-benar merupakan kerentanan yang valid dan bukan merupakan kesalahan deteksi (*false positive*). Tahap ini dilakukan dengan mencoba melakukan injeksi *payload* sederhana untuk melihat respon aplikasi secara real (Teguh Yuwono et al., 2021)
4. Risk Analysis & Assessment: Setelah kerentanan dikonfirmasi dilakukan penilaian terhadap tingkat risiko yang ditimbulkan. Penilaian ini biasanya mengacu pada standar skor keparahan untuk menentukan dampak eksploitasi terhadap integritas dan kerahasiaan data jurnal (Riadi et al., 2020)

Implementasi metodologi VA pada platform OJS sangat krusial, mengingat sifat platform yang *open-source* dan memiliki banyak titik input data yang kompleks. Dengan melakukan audit keamanan secara berkala menggunakan metodologi VA institusi pengelola jurnal dapat merancang strategi mitigasi yang tepat dan efisien berdasarkan temuan celah keamanan yang paling berisiko (Artikel, 2023)

2.2.11 Tools Pengujian

XSSer (singkatan dari *Cross Site Scripter*) adalah sebuah *framework open source* tingkat lanjut yang dirancang khusus untuk mendeteksi, mengeksploitasi, dan melaporkan kerentanan *Cross-Site Scripting* (XSS) pada aplikasi berbasis web. Dalam penelitian ini digunakan XSSer versi 1.8.4 yang merupakan salah satu alat penetrasi otomatis paling komprehensif untuk pengujian XSS karena menyediakan berbagai modul serangan terintegrasi (Saputra & Hidayat, 2025).

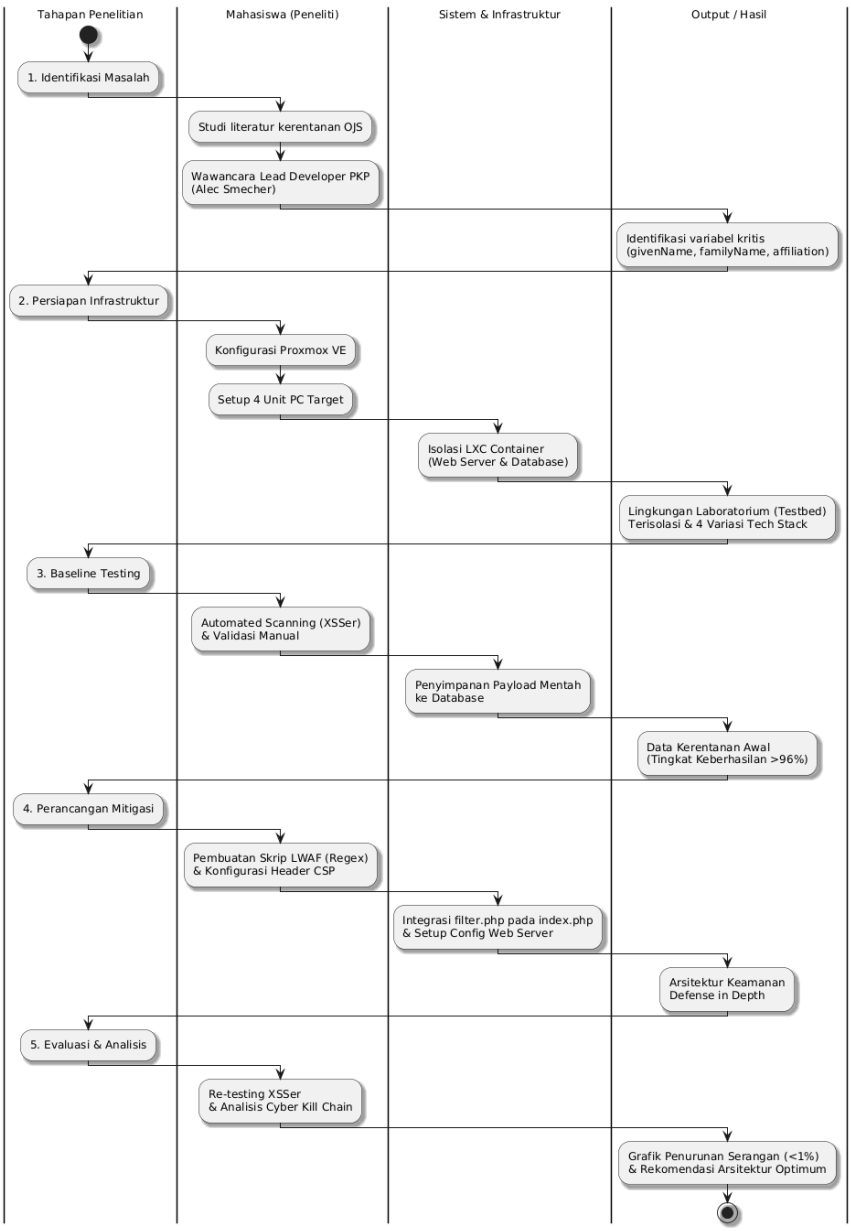
XSSer bekerja dengan mengotomatisasi proses penyuntikan *payload* ke dalam berbagai titik input pada platform OJS. Beberapa fitur dan mekanisme teknis utama dari XSSer v1.8.4 yang digunakan dalam penelitian ini meliputi:

1. Heuristic Engine: Fitur ini memungkinkan XSSer untuk melakukan analisis awal terhadap respon server guna memetakan parameter mana saja yang memiliki potensi kerentanan tanpa harus mengirimkan seluruh database *payload*. Mesin ini mendeteksi apakah karakter khusus tertentu (seperti < > " ') dipantulkan kembali ke browser atau difilter oleh server (Babaey & Ravindran, 2025)
2. Injection Techniques: XSSer mendukung otomatisasi serangan untuk tiga kategori utama XSS: *Stored, Reflected,* dan *DOM-based*. Alat ini mampu mengirimkan ribuan variasi skrip jahat ke berbagai *endpoint* OJS mulai dari formulir metadata hingga parameter pencarian URL (Babaey & Ravindran, 2025)
3. Bypassing and Encoding: Salah satu keunggulan teknis XSSer v1.8.4 adalah kemampuannya untuk mengelabui filter keamanan sederhana melalui teknik *encoding*. XSSer dapat mengubah *payload* standar menjadi format lain seperti *Hexadecimal, Octal, Unicode,* atau *Double-URL Encoding*. Teknik ini sangat krusial untuk mengevaluasi ketangguhan arsitektur mitigasi dalam mendeteksi serangan yang telah dimodifikasi (Weamie, 2022)
4. Crawl & Discovery: XSSer dilengkapi dengan fungsi *crawler* untuk melakukan pemetaan otomatis terhadap seluruh tautan internal pada platform OJS memastikan tidak ada parameter input yang terlewatkan selama proses *Vulnerability Assessment*.

Penggunaan XSSer dalam skripsi ini bertujuan untuk menghasilkan data pengujian yang objektif dan terukur. Dengan memanfaatkan otomatisasi yang disediakan oleh XSSer peneliti dapat membandingkan efektivitas mitigasi WAF pada PC 1 hingga PC 4 secara konsisten menggunakan kumpulan *payload* yang sama (A. F. Lufna et al., 2025).

2.3 Kerangka Pemikiran

Adapaun kerangka pemikiran pada penelitian ini adalah sebagai berikut:



Gambar 2. 5 Kerangka Pemikiran

Gambar 2.5 menunjukkan kerangka pemikiran yang mendasari alur penelitian ini dari awal hingga akhir. Proses ini dibagi menjadi empat pilar utama untuk menunjukkan interaksi antara tahapan formal, aktivitas peneliti, keterlibatan sistem, dan hasil yang diperoleh. Penjabaran dari kerangka pemikiran tersebut adalah sebagai berikut:

1. Tahap Identifikasi Masalah

Penelitian dimulai dengan studi literatur dan diperkuat melalui wawancara dengan Alec Smecher. Aktivitas ini bertujuan untuk melakukan observasi empiris terhadap kerentanan aplikasi OJS. *Output* dari tahap ini adalah identifikasi variabel input kritis pada form registrasi yaitu *givenName, familyName,* dan *affiliation* yang akan dijadikan parameter pengujian.

2. Tahap Persiapan Infrastruktur

Pada tahap ini peneliti membangun lingkungan laboratorium yang terisolasi. Peneliti melakukan konfigurasi *hypervisor* Proxmox VE dan melakukan instalasi 4 Kombinasi unit PC target. Sistem diatur menggunakan isolasi *LXC Container* untuk memisahkan *Web Server* dan *Database* untuk menerapkan prinsip *segregation of duties*. Hasilnya adalah ekosistem infrastruktur yang siap diuji dengan berbagai kombinasi arsitektur.

3. Tahap *Baseline Testing*

Tahap ini bertujuan untuk memetakan kerentanan awal sistem dalam kondisi standar. Peneliti melakukan simulasi serangan menggunakan *tools* XSSer (otomatis) dan validasi manual. Secara teknis sistem akan menerima *payload* mentah dan menyimpannya ke dalam basis data. Hasil pengujian ini menyediakan data dasar (*baseline*) yang menunjukkan tingkat keberhasilan serangan yang sangat tinggi (>96%).

4. Tahap Perancangan Mitigasi

Berdasarkan data kerentanan awal peneliti merancang arsitektur keamanan *Defense in Depth*. Aktivitas ini mencakup pembuatan skrip LWAF berbasis *Regular Expression* (Regex) dan konfigurasi *header* CSP pada *Web Server*. Secara sistem mitigasi ini diintegrasikan pada file index.php (sebagai *Front Controller*). *Output*-nya adalah sebuah model perlindungan berlapis yang mengamankan sisi server dan sisi klien.

5. Tahap Evaluasi & Analisis

Tahap akhir adalah melakukan pengujian ulang (*re-testing*) untuk mengukur efektivitas mitigasi. Peneliti menganalisis hasil serangan menggunakan metodologi *Cyber Kill Chain* untuk melihat pada tahap mana rantai serangan berhasil diputus. Tahap ini menghasilkan

grafik perbandingan hasil yang signifikan (penurunan serangan hingga <1%) dan merumuskan rekomendasi arsitektur infrastruktur paling optimum bagi pengelola jurnal.