Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

ANALISIS KERENTANAN SQLi DAN XSS PADA WEBSITE TOP-UP GAME

Heri Sumantri[1], Husain[2], Muhamad Azwar[3], Raisul Azhar[4], Khairan Marzuki[5]

[123]Program Studi Teknologi Informasi Universitas Bumigora, [45]ProgramStudi Ilmu Komputer Universitas Bumigora

Jln. Ismail Marzuki No.22, Universitas Bumigora, Cilinaya, Cakranegara, Mataram 83127
[1] ftikaboiooherisumantri@gmail.com, [2] husain@universitasbumigora.ac.id ,
[3] azwar@universitasbumigora.ac.id, [4]raisulazhar@universitasbumigora.ac.id,
[5]khairan.marzuki@universitasbumigora.ac.id

Abstract

SQL Injection and Cross-Site Scripting (XSS) consistently rank as the most critical vulnerabilities in web applications according to the OWASP Top 10 2021, with 94% of web applications at risk of intrusion. This study aims to analyze SQL Injection and XSS vulnerabilities on the LisxDragon website, a top-up gaming platform in Indonesia, using a white-box penetration testing methodology based on the OWASP Web Security Testing Guide (WSTG). Out of 122 mapped endpoints, 6 valid vulnerabilities were identified across 2 endpoints: four variants of SQL Injection (Error-Based, Time-Based, Boolean-Based, Union-Based) and Reflected XSS. Controlled exploitation confirmed tangible impacts including database structure leaks, service disruption, query logic manipulation, user data extraction, and session cookie theft. All vulnerabilities were successfully mitigated using prepared statements and HTML special character encoding (htmlspecialchars()), with a post-mitigation rescan confirming that all endpoints are now secure.

Keywords : *SQL Injection, Cross-Site Scripting, Penetration Testing, Owasp, Web Security*

Abstrak

SQL Injection dan Cross-Site Scripting (XSS) secara konsisten menduduki peringkat sebagai kerentanan paling kritis dalam aplikasi web menurut OWASP Top 10 2021, dengan 94% aplikasi web berisiko mengalami intrusi. Penelitian ini bertujuan menganalisis kerentanan SQL Injection dan XSS pada situs web LisxDragon, sebuah platform *game top-up* di Indonesia, menggunakan metodologi *white-box penetration testing* berbasis OWASP Web Security Testing Guide (WSTG). Dari 122 endpoint yang dipetakan, ditemukan 6 kerentanan valid pada 2 endpoint: empat varian SQL Injection (*Error-Based, Time-Based, Boolean-Based, Union-Based*) dan *Reflected XSS*. Eksploitasi terkontrol mengonfirmasi dampak nyata berupa kebocoran struktur basis data, gangguan layanan, manipulasi logika kueri, ekstraksi data pengguna, dan pencurian cookie sesi. Seluruh kerentanan berhasil dimitigasi menggunakan *prepared statement* dan *output encoding* htmlspecialchars(), dengan rescan pasca-mitigasi yang mengonfirmasi seluruh endpoint kini berstatus aman.

Kata kunci : *SQL Injection, Cross-Site Scripting, Penetration Testing, Owasp, Keamanan Web*

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

1. PENDAHULUAN

Perkembangan pesat teknologi informasi telah mendorong munculnya platform digital berbasis web, yang paling menonjol di antaranya adalah layanan top-up game online yang memungkinkan pengguna melakukan pembelian mata uang virtual, item digital, atau layanan dalam permainan secara daring. Platform ini menyimpan dan memproses data sensitif pengguna, termasuk informasi akun, riwayat transaksi, dan rincian pembayaran digital. Seiring meningkatnya jumlah transaksi, platform top-up game menjadi sasaran yang lebih menarik bagi kejahatan siber, yang berdampak pada aset digital yang dibeli dan data keuangan yang disimpan di dalamnya.

Ancaman keamanan siber di Indonesia terus mengalami peningkatan. Badan Siber dan Sandi Negara (BSSN) menegaskan bahwa serangan terhadap aplikasi web, termasuk SQL Injection, masih menjadi salah satu perhatian penting dalam penanganan insiden keamanan siber di Indonesia [1]. Anomali tersebut mencakup berbagai jenis serangan terhadap infrastruktur digital, termasuk serangan yang menargetkan lapisan aplikasi web. Kondisi ini mengindikasikan bahwa transformasi digital yang tidak diiringi dengan penguatan keamanan sistem akan menciptakan celah yang dapat dieksploitasi oleh pihak yang tidak bertanggung jawab.

Di antara berbagai ancaman yang ada, SQL Injection dan *Cross-Site Scripting* (XSS) merupakan kerentanan yang paling sering ditemukan dan paling berdampak pada aplikasi web. Open Web Application Security Project (OWASP) dalam dokumen Top 10 2021 mengklasifikasikan injeksi sebagai A03:2021 — risiko paling kritis — dengan catatan bahwa 94% aplikasi web yang diuji berpotensi mengalami intrusi melalui celah injeksi, dan total 274.228 insiden dilaporkan secara global [2]. Angka ini menegaskan bahwa kerentanan injeksi bukan sekadar ancaman teoretis, melainkan masalah nyata yang secara aktif dieksploitasi.

Relevansi isu ini di Indonesia tercermin dari kejadian yang diberitakan pada 4 Maret 2025, ketika situs web resmi Badan Narkotika Nasional (BNN) mengalami defacement akibat serangan SQL injection. Laporan Cyber Press menyebutkan bahwa serangan tersebut dilakukan oleh pelaku dengan alias "Havij Santana", yang Merujuk pada alat Havij SQL injection, serta mengindikasikan adanya akses tidak sah terhadap basis data internal BNN [3]. Kasus serupa berpotensi lebih sering terjadi pada platform komersial seperti layanan *top-up game* yang umumnya dikelola oleh pengembang dengan sumber daya keamanan yang lebih terbatas dibandingkan institusi besar.

SQL Injection adalah teknik serangan yang memanfaatkan kelemahan validasi masukan pada aplikasi web sehingga penyerang dapat menyisipkan perintah SQL berbahaya ke dalam parameter input yang diproses server, sehingga penyerang dapat memanipulasi, mengakses, mengubah, dan menghapus data dalam basis data tanpa otorisasi yang sah. Menurut Bastian [4], hal ini terjadi karena proses pengembangan melibatkan konversi dinamis masukan pengguna menjadi string SQL tanpa menggunakan mekanisme parameterisasi.

Cross-Site Scripting (XSS) salah satu bentuk serangan pada aplikasi web. [5] XSS adalah bentuk serangan injeksi. Serangan injeksi dilakukan dengan menyisipkan kode javascript melalui fasilitas interaksi yang diberikan oleh aplikasi web kepada pengguna. Serangan XSS yang berhasil dilakukan dapat mengakibatkan pelanggaran yang serius baik pada website itu sendiri maupun pada pengguna lain. Peretas dapat menginjeksikan kode berbahaya melalui aplikasi web yang mengijinkan pengguna melakukan masukan yang tidak divalidasi dengan baik. Akibat dari serangan tersebut peretas dapat mencuri cookie, mengambil informasi yang sifatnya pribadi, membajak akun pengguna, memanipulasi konten web, melakukan serangan danial of service dan berbagai macam aktifitas berbahaya lainnya.

Penetration testing adalah metode pengujian keamanan yang mensimulasikan serangan dunia nyata terhadap suatu sistem untuk mengidentifikasi dan mengevaluasi kerentanan yang ada. [6] *penetrasi testing* metode untuk menilai kerentanan pada sistem, Pengujian ini

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

untuk mengidentifikasi masalah dalam sistem informasi yang sedang diperiksa. Tujuan utama dari *penetration testing* adalah untuk mengidentifikasi dan mengatasi kelemahan keamanan dalam infrastruktur jaringan.

LisxDragon adalah platform digital yang menyediakan layanan *top-up game online* dan pembayaran prabayar secara daring. Platform ini menyimpan informasi sensitif berupa akun pengguna, data transaksi, serta detail pembayaran — seluruhnya menjadi target potensial jika terjadi pelanggaran keamanan. Meskipun platform *top-up game* semakin populer di Indonesia seiring pertumbuhan industri game digital, kajian keamanan yang spesifik menyasar jenis platform ini masih sangat terbatas.

Berbeda dengan sebagian besar penelitian keamanan web yang berfokus pada identifikasi kerentanan dan rekomendasi perbaikan, penelitian ini mengintegrasikan proses identifikasi, eksploitasi, mitigasi, dan verifikasi ulang dalam satu alur pengujian yang utuh. Selain itu, penelitian ini menerapkan pendekatan white-box penetration testing berbasis OWASP WSTG pada platform top-up game, yang hingga saat ini masih relatif jarang menjadi objek penelitian keamanan. Dengan demikian, penelitian ini tidak hanya mengidentifikasi keberadaan kerentanan, tetapi juga membuktikan dampaknya, menerapkan perbaikan pada level kode sumber, serta memverifikasi efektivitas mitigasi yang dilakukan.

Berdasarkan latar belakang tersebut, penelitian ini bertujuan untuk: (1) mengidentifikasi dan menganalisis kerentanan SQL Injection dan XSS pada website LisxDragon secara sistematis menggunakan metodologi *white-box penetration testing* (2) membuktikan dampak nyata kerentanan melalui eksploitasi terkontrol sebagai *proof-of-concept* dengan pendukung pengujian OWASP WSTG serta (3) memberikan mitigasi yang diimplementasikan langsung pada level kode sumber dan diverifikasi melalui pengujian ulang (*rescan*).

Hasil penelitian berkontribusi pada pengembangan sistem atau situs web administrator LisxDragon, Sebagai penanggung jawab aplikasi web. Berdasarkan hasil analisis dan pengujian, penelitian ini membantu mereka mengidentifikasi dan memperbaiki kerentanan kritis seperti SQL injection dan Cross Site Scripting (XSS). Dan diharapkan menjadi referensi bagi pengelola platform web serupa dalam melaksanakan audit keamanan yang terstruktur dan efektif.

2. TINJAUAN PUSTAKA

Tinjauan terhadap penelitian terdahulu dilakukan untuk mengidentifikasi pendekatan yang telah digunakan serta menemukan celah penelitian (*research gap*) yang dapat dikembangkan lebih lanjut dalam studi ini.

Menurut penelitian [7] mengevaluasi keamanan situs web menggunakan metode OWASP, dengan fokus pada SQL Injection dan XSS. Penelitian ini berhasil mengidentifikasi beberapa kerentanan dengan tingkat risiko yang semakin tinggi, serta memberikan rekomendasi perbaikan seperti validasi input dan manajemen sesi. Namun, penelitian ini memiliki keterbatasan karena kurangnya data berkualitas tinggi dan keterbatasan dalam melakukan analisis data secara menyeluruh (proof-of-concept).

Menurut penelitian [8] melakukan penetrasi pada situs web STIE Samarinda menggunakan SQL Injection dan XSS. Temuan penelitian ini mengungkapkan 14 ambang kerentanan yang dapat dieksploitasi, terutama pada berbagai formulir input. Meskipun penelitian ini cukup komprehensif dalam hal mengidentifikasi dan menghilangkan kelemahan, keterbatasan utamanya adalah kurangnya implementasi perbaikan yang sistematis terhadap sistem yang diteliti.

Menurut penelitian [9] menganalisis keamanan sebuah situs web terhadap serangan SQL Injection dan XSS dengan menggunakan kombinasi metode manual dan otomatis. Hasil penelitian menunjukkan bahwa situs web

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

tersebut rentan terhadap serangan SQL Injection, namun tidak ditemukan bukti adanya serangan XSS karena mekanisme keamanan seperti Content Security Policy (CSP) telah diterapkan. Keterbatasan penelitian ini terletak pada cakupan yang sempit, di mana analisis lebih berfokus pada SQL Injection daripada XSS dan tidak mencakup semua variasi yang mungkin terjadi.

Menurut penelitian [10], efektivitas OWASP ZAP dalam mendeteksi kerentanan SQL Injection dianalisis menggunakan lingkungan pengujian OWASP Juice Shop dan Damn Vulnerable Web Application (DVWA). Hasil penelitian menunjukkan bahwa OWASP ZAP mampu mendeteksi kerentanan yang menghasilkan respons eksplisit, seperti Error-Based SQL Injection, namun masih memiliki keterbatasan dalam mengidentifikasi beberapa variasi Blind SQL Injection sehingga diperlukan validasi dan pengujian manual untuk memperoleh hasil yang lebih komprehensif. Keterbatasan penelitian ini terletak pada fokusnya yang hanya mengevaluasi kemampuan deteksi alat pengujian tanpa membahas implementasi mitigasi maupun verifikasi ulang setelah perbaikan dilakukan.

Menurut penelitian [11], deteksi serangan SQL Injection dan Cross-Site Scripting (XSS) dapat ditingkatkan melalui pendekatan machine learning yang menggabungkan beberapa teknik ekstraksi fitur, seperti Word2Vec, Universal Sentence Encoder (USE), dan FastText. Hasil penelitian menunjukkan bahwa pendekatan tersebut mampu meningkatkan akurasi deteksi terhadap berbagai pola serangan pada aplikasi web. Namun, penelitian ini berfokus pada pengembangan model deteksi berbasis machine learning dan tidak membahas proses eksploitasi, implementasi mitigasi, maupun verifikasi ulang terhadap kerentanan yang ditemukan pada aplikasi web.

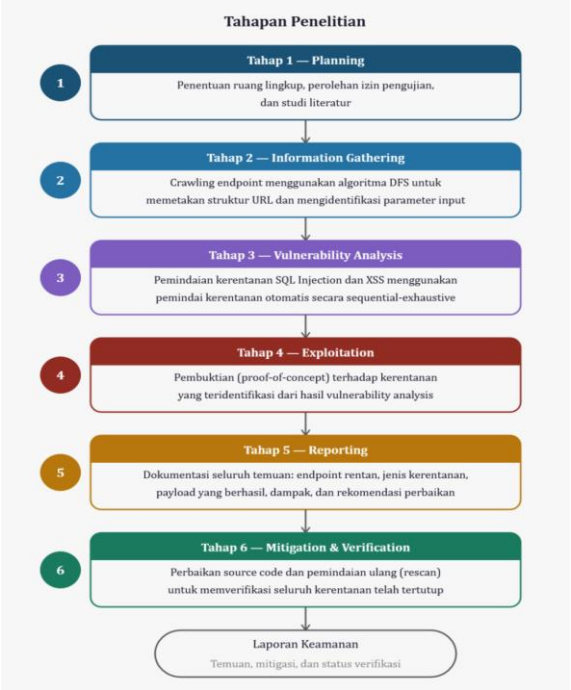
Berdasarkan kajian tersebut, terdapat tiga celah penelitian utama, yaitu: (1) sebagian besar penelitian masih berfokus pada tahap identifikasi kerentanan tanpa melakukan validasi eksploitasi secara menyeluruh terhadap setiap temuan; (2) belum banyak penelitian yang mengintegrasikan proses identifikasi, eksploitasi, mitigasi, dan verifikasi ulang dalam satu alur pengujian yang komprehensif; serta (3) masih terbatasnya kajian keamanan yang secara khusus meneliti platform top-up game sebagai objek penelitian. Oleh karena itu, penelitian ini mengusulkan pendekatan white-box penetration testing berbasis OWASP WSTG yang mencakup pemetaan endpoint menggunakan Depth First Search (DFS), pengujian sequential-exhaustive, pembuktian eksploitasi (proof-of-concept), implementasi mitigasi pada level kode sumber, dan verifikasi pasca-perbaikan.

3. METODOLOGI PENELITIAN

3.1. Tahapan Penelitian

Penelitian ini menggunakan metode *penetration testing* sebagai pendekatan utama dalam pengujian keamanan aplikasi web. *Penetration testing* adalah metode pengujian yang mensimulasikan serangan nyata terhadap sistem secara terstruktur dan legal untuk mengidentifikasi, membuktikan, serta membantu memperbaiki kerentanan keamanan sebelum dieksploitasi oleh pihak yang tidak bertanggung jawab berdasarkan [6]. OWASP Web Security Testing Guide (WSTG) yang merupakan standar internasional untuk pengujian keamanan aplikasi web sebagai acuan atau pendukung penelitian ini [12]. Tahapan penelitian terdiri dari enam fase berurutan sebagaimana diilustrasikan pada Gambar 1.

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi



Gambar 1. Tahapan Penelitian Dan Penetrasi Testing

3.2. Pengumpulan Data

Penelitian ini menggunakan dua jenis data. Data primer diperoleh langsung melalui proses *penetration testing* pada website LisxDragon yang mencakup: (a) hasil pemindaian endpoint menggunakan scanner dengan pendekatan *DFS*, *sequential-exhaustive*; (b) respons HTTP server terhadap setiap payload injeksi; dan (c) dokumentasi eksploitasi berupa *screenshot* hasil serangan terkontrol. Seluruh pengujian dilaksanakan atas izin tertulis dari pemilik website sebagai prasyarat utama *penetration testing* [6]. Kode sumber (*source code*) website juga menjadi data primer dalam pendekatan *white-box* untuk analisis akar penyebab kerentanan dan implementasi mitigasi.

Data sekunder diperoleh dari studi pustaka terhadap standar dan literatur yang relevan, yaitu: OWASP Web Security Testing Guide (WSTG) sebagai acuan metodologi pengujian [12] ; OWASP Top 10 2021 sebagai referensi klasifikasi kerentanan [2]; laporan BSSN terkait insiden siber di Indonesia [1]; serta penelitian terdahulu yang berkaitan dengan SQL Injection, XSS, dan *penetration testing* pada aplikasi web [4],[5],[6].

3.3. Analisa Data

Analisis data dilakukan menggunakan pendekatan 5W1H terhadap kerentanan yang ditemukan pada website LisxDragon:

Why (Mengapa): Website LisxDragon menyimpan data sensitif pengguna berupa akun, riwayat transaksi, dan detail pembayaran. Ketiadaan validasi input pada parameter kritis menjadi penyebab utama kerentanan SQL Injection dan XSS yang ditemukan.

Who (Siapa): Pihak yang terdampak adalah pengguna aktif platform LisxDragon yang data dan sesi loginnya berisiko bocor atau dicuri. Pengujian dilakukan oleh peneliti atas izin pemilik website.

What (Apa): Kerentanan yang dianalisis adalah SQL Injection (*Error-Based, Time-Based, Boolean-Based, Union-Based*) dan *Reflected XSS*

Where (Di mana): Kerentanan yang ditemukan pada endpoint yang teridentifikasi.

When (Kapan): Pengujian dilakukan setelah izin tertulis diperoleh, mencakup enam fase berurutan: perencanaan, pengumpulan informasi, pemindaian, eksploitasi, pelaporan, dan mitigasi dengan verifikasi ulang.

How (Bagaimana): Analisis dilakukan dengan membandingkan respons HTTP server terhadap *baseline* melalui lima mekanisme deteksi: identifikasi pesan error database, pengukuran delta waktu respons, perbandingan kondisi TRUE/FALSE, deteksi *signature* UNION, dan pemeriksaan refleksi payload XSS tanpa *encoding*.

4. HASIL DAN PEMBAHASAN

4.1 Planning

Pada fase *planning*, peneliti menetapkan ruang lingkup pengujian yang mencakup seluruh domain Lisxdragon dengan fokus pada deteksi kerentanan SQL Injection dan XSS. Izin pengujian diperoleh secara tertulis dari pemilik website

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

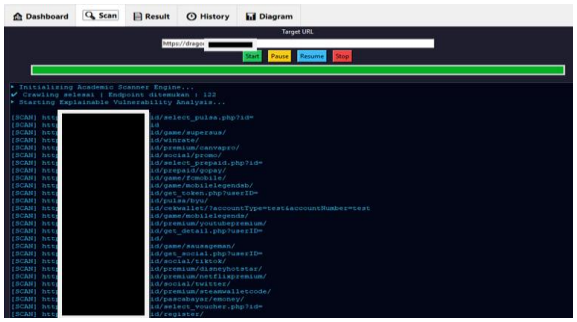
sebagai prasyarat legal *penetration testing* mengacu pada [6]. Studi literatur dilakukan terhadap OWASP WSTG sebagai acuan pengujian pada [12], OWASP Top 10 2021 sebagai referensi klasifikasi kerentanan [2], serta penelitian terdahulu yang relevan [7],[8],[9]. Pendekatan yang ditetapkan adalah *white-box penetration testing* karena peneliti memiliki akses penuh terhadap kode sumber aplikasi, sehingga analisis akar penyebab kerentanan dapat dilakukan secara mendalam.

4.2 Information Gathering

Fase *information gathering* dilakukan menggunakan scanner algoritma *Depth First Search* (DFS) untuk menelusuri seluruh struktur URL website secara rekursif. DFS adalah pencarian yang berjalan dengan meluaskan anak akar pertama dari pohon pencarian yang dipilih dan berjalan dalam dan lebih dalam lagi sampai simpul tujuan ditemukan, atau sampai menemukan simpul yang tidak punya anak. Kemudian, pencarian *backtracking*, akan kembali kesimpul yang belum selesai ditelusuri [13].

Hasil fase ini adalah peta seluruh endpoint website yang menjadi cakupan pengujian,

Total **122 endpoint** berhasil dipetakan dan menjadi cakupan pengujian pada fase berikutnya.



Gambar 2. Hasil Crawling Endpoint

4.3 Vulnerability Analysis

Fase *vulnerability analysis* dilakukan menggunakan scanner dengan pendekatan *sequential-exhaustive*: algoritma *sequential* pencarian data yang dilakukan secara berurutan dari awal hingga akhir atau dari depan ke belakang, aliran pencarian berurutan, proses pencarian data melibatkan perbandingan data satu per satu dengan data awal hingga akhir aliran [14]. Algoritma *exhaustive* memeriksa sasecara sistematis setiap kemungkinan solusi satu per satu dalam pencarian solusinya. Meskipun algoritma *Exhaustive* Search secara teoriti smenghasilkan solusi mengacu pada [15]. Artinya *sequential* setiap endpoint diuji satu per satu, dan *exhaustive* setiap parameter diuji terhadap seluruh kategori payload SQLi dan XSS tanpa pengecualian. Dari 122 endpoint yang dipindai, scanner mengidentifikasi **6 kerentanan valid pada 2 endpoint** sebagaimana disajikan pada Tabel 1 dan Gambar 3,4,5,6.

Tabel 1. Ringkasan Kerentanan Website LisxDragon

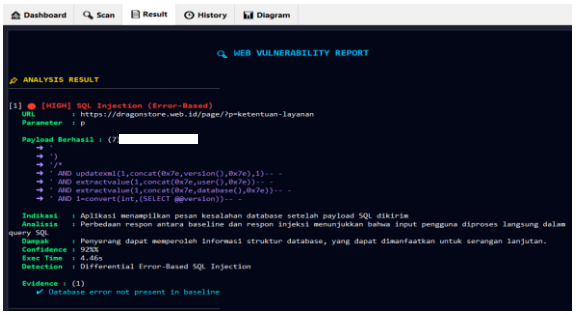
No	Jenis Kerentanan	Endpoint	Param
1	SQLi Error-Based	/page/?p=ketentuan-layanan	p
2	SQLi Time-Based	/page/?p=ketentuan-layanan	p
3	SQLi Time-Based	/cektrx/?trxNum=	trxNum
4	SQLi Boolean-Based	/cektrx/?trxNum=	trxNum
5	SQLi Union-Based	/cektrx/?trxNum=	trxNum
6	XSS Reflected	/cektrx/?trxNum=	trxNum

Endpoint */cektrx/?trxNum=* (fitur Cek Status Pesanan) menunjukkan variasi kerentanan paling dominan dengan empat dari enam temuan berada pada satu endpoint. Kondisi ini disebabkan karena parameter trxNum menerima nilai dari input pengguna dan langsung disisipkan ke dalam kueri SQL tanpa mekanisme parameterisasi, sehingga seluruh teknik injeksi yang mengandalkan manipulasi sintaks SQL berhasil dieksekusi. Menurut penelitian [4] menjelaskan bahwa pola konstruksi kueri dinamis semacam ini adalah penyebab paling umum kerentanan SQL Injection pada aplikasi berbasis PHP-MySQL.

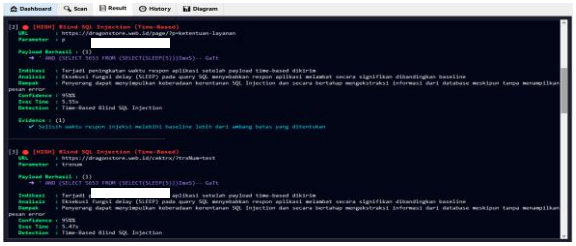
Endpoint */page/?p=* rentan terhadap dua varian SQL Injection akibat pola yang sama pada konstruksi kueri di sisi server. Meskipun hanya dua varian yang terdeteksi, kerentanan *Error-*

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

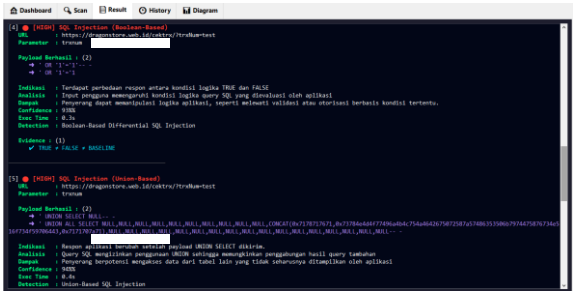
Based pada endpoint ini berpotensi memberikan informasi struktur basis data secara lengkap kepada penyerang melalui pesan kesalahan yang diekspos langsung pada halaman respons.



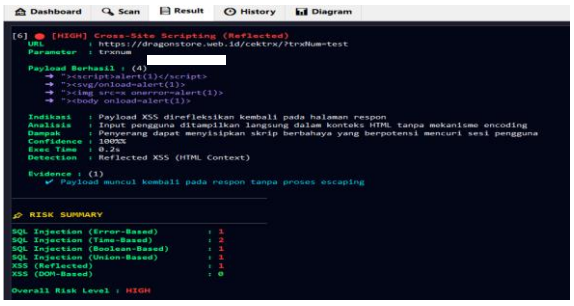
Gambar 3. Hasil Vulnerability Analysis SQLi Error-Based



Gambar 4. Hasil Vulnerability Analysis SQLi Time-Based



Gambar 5. Hasil Vulnerability Analysis SQLi Boolean-Based Dan Union-Based



Gambar 6. Hasil Vulnerability Analysis XSS Reflected

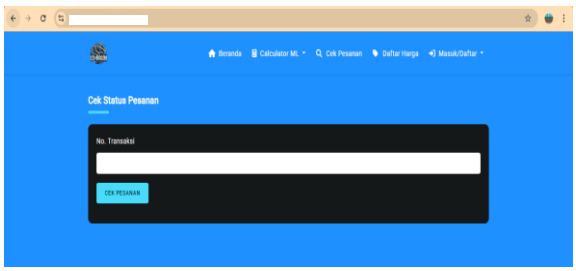
Pada proses identifikasi kerentanan SQLi Union-Based dan Time-Based, deteksi awal dilakukan oleh pemindai kerentanan otomatis dengan pendekatan DFS sequential-exhaustive, kemudian diperkuat menggunakan payload SQLMap untuk memverifikasi akurasi hasil dan mengonfirmasi bahwa kerentanan dapat dieksploitasi secara nyata.

4.4 Exploitation

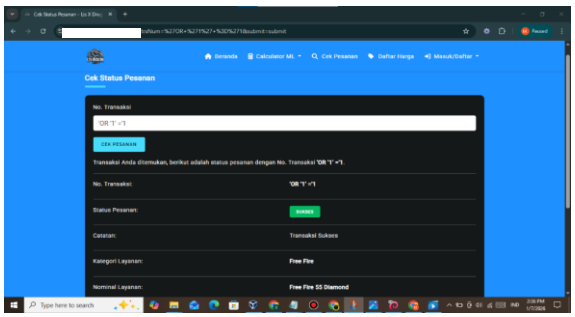
Fase eksploitasi dilakukan sebagai pembuktian (*proof-of-concept*) terhadap seluruh kerentanan yang teridentifikasi pada fase scanning. Tujuannya adalah mengonfirmasi *true positive* dan mengukur dampak nyata yang dapat ditimbulkan jika kerentanan dieksploitasi oleh penyerang [4],[12].

4.4.1 SQL Injection Boolean-Based

Pengujian *Boolean-Based* dengan payload *' OR '1'='1* pada parameter trxNum menghasilkan perbedaan respons signifikan dibandingkan *baseline*: kondisi TRUE mengembalikan data transaksi, kondisi FALSE tidak. Perbedaan ini membuktikan bahwa logika klausa WHERE dapat dimanipulasi melalui input pengguna — indikasi bahwa input tidak diparameterisasi berdasarkan [16],[12].



Gambar 7. Respon Baseline Pada Endpoint /cektrx/?trxNum=

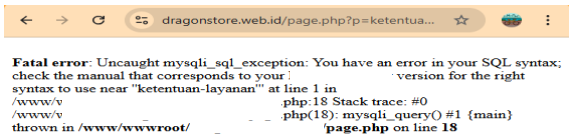


Gambar 8. Respon Pengujian SQLi Boolean-Based Pada Endpoint /cektrx/?trxNum=

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

4.4.2 SQL Injection Error-Based

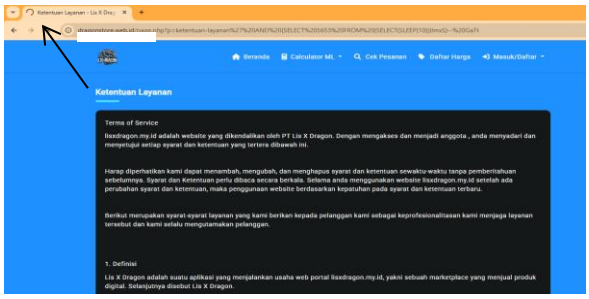
Pengujian *Error-Based* dengan menyisipkan karakter kutip tunggal pada parameter p memunculkan pesan kesalahan database yang mengandung informasi teknis sensitif — jenis DBMS dan struktur kueri. Menurut [17],[12] menegaskan bahwa ekspos pesan kesalahan ini secara langsung memberikan informasi kepada penyerang untuk merancang serangan lebih presisi.



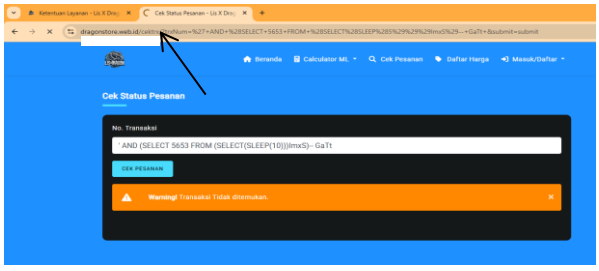
Gambar 9. Respon Pengujian SQLi Error-Based Pada Endpoint /page/?p=ketentuan-layanan

4.4.3 SQL Injection Time-Based

Pengujian *Time-Based* menghasilkan jeda respons >5 detik pada kedua endpoint mengonfirmasi bahwa fungsi SLEEP() berhasil dieksekusi oleh mesin basis data [4],[18]. teknik ini efektif untuk mendeteksi kerentanan *blind SQL Injection* di mana tidak ada perbedaan respons visual.



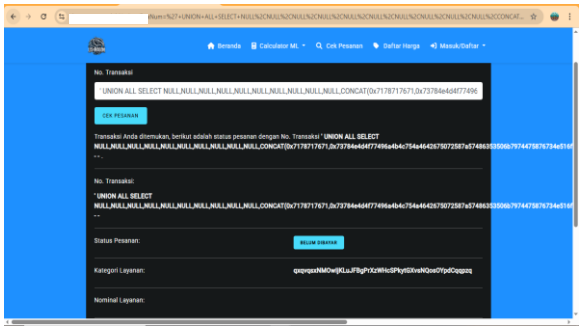
Gambar 10. Respon Pengujian Time-Based Pada Endpoint /page/?p=ketentuan-layanan



Gambar 11. Respon Pengujian SQLi Time-Based Pada Endpoint /cektrx/?trxNum=

4.4.4 SQL Injection Union-Based

Pengujian *Union-Based* menggunakan payload UNION SELECT dengan *signature* unik berhasil menampilkan *signature* tersebut dalam respons HTML — mengindikasikan data dari tabel lain dapat dibaca. Eksploitasi lanjutan mengonfirmasi bahwa struktur tabel dan isi data pengguna (tabel tb_user) dapat diekstraksi sepenuhnya [12].

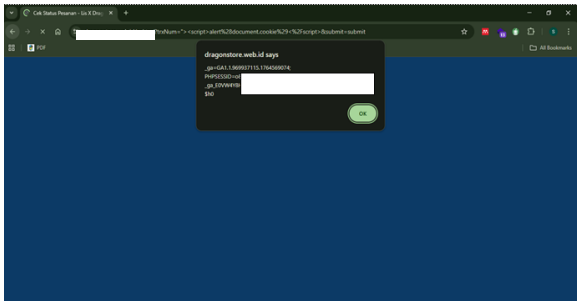


Gambar 12. Respon Pengujian SQLi Union-Based Pada Endpoint /cektrx/?trxNum=

4.4.5 XSS Reflected

Pengujian Reflected XSS menggunakan payload injeksi script pada parameter trxNum berhasil dieksekusi browser — membuktikan nilai parameter dikembalikan ke HTML tanpa output encoding [12]. Xss reflected terjadi ketika seorang pengguna mengirimkan input ke suatu aplikasi dan kemudian input tersebut dikembalikan kepada pengguna yang sama. Skrip dieksekusi melalui koneksi yang mengirimkan permintaan ke server web jarak jauh yang mampu mengeksekusi skrip tersebut [19]. Pengujian lanjutan menggunakan payload yang menampilkan cookie sesi berhasil mengekspos nilai cookie sesi pengguna — mengonfirmasi risiko *session hijacking* jika URL berbahaya disebarkan kepada pengguna lain.

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi



Gambar 13. Respon Pengujian XSS Reflected Pada Endpoint /cektrx/?trxNum=

4.5 Reporting

Berdasarkan hasil fase *vulnerability analysis* dan eksploitasi, seluruh temuan didokumentasikan secara terstruktur. Ditemukan 6 kerentanan valid pada 2 endpoint — seluruhnya masuk dalam Owasp 10 [2]. Dampak yang terkonfirmasi dari fase eksploitasi mencakup: (1) kebocoran struktur basis data dan informasi DBMS melalui SQLi *Error-Based*; (2) ekstraksi data seluruh tabel pengguna melalui SQLi *Union-Based*; (3) manipulasi logika kueri untuk mengakses data tanpa autentikasi melalui SQLi *Boolean-Based*; (4) gangguan ketersediaan layanan melalui eksekusi fungsi penundaan pada SQLi *Time-Based*; dan (5) pencurian cookie sesi yang memungkinkan pembajakan akun pengguna melalui XSS Reflected.

Temuan ini mengonfirmasi bahwa kedua endpoint rentan menghadapi risiko keamanan yang serius dan memerlukan perbaikan segera pada level kode sumber. Seluruh dokumentasi temuan, termasuk *payload* yang berhasil, respons server, dan dampak eksploitasi, menjadi dasar rekomendasi mitigasi yang diterapkan pada fase berikutnya.

Berdasarkan tingkat dampak yang ditimbulkan, SQL Injection Union-Based dan Reflected XSS merupakan kerentanan dengan tingkat risiko tertinggi karena memungkinkan akses langsung terhadap data sensitif dan sesi pengguna. SQL Injection Union-Based terbukti mampu mengekstraksi informasi dari tabel pengguna sehingga berpotensi menyebabkan kebocoran data dalam skala besar. Reflected XSS juga memiliki tingkat risiko tinggi karena memungkinkan pencurian cookie sesi yang dapat dimanfaatkan untuk pengambilalihan akun pengguna. SQL Injection Error-Based dikategorikan sebagai risiko tinggi karena mengungkap informasi internal basis data yang dapat digunakan untuk menyusun serangan lanjutan. Sementara itu, SQL Injection Boolean-Based dan Time-Based berada pada tingkat risiko sedang hingga tinggi karena meskipun tidak secara langsung menampilkan data sensitif, keduanya tetap memungkinkan manipulasi logika kueri dan identifikasi kerentanan yang dapat dimanfaatkan pada tahap eksploitasi berikutnya. Temuan ini menunjukkan bahwa kerentanan yang berdampak langsung terhadap kerahasiaan data dan sesi pengguna harus menjadi prioritas utama dalam proses mitigasi.

4.6 Mitigation & Verification

4.6.1 Mitigasi SQL Injection Prepared Statement

Mitigasi SQL Injection dilakukan dengan menerapkan *prepared statement* (*parameterized query*) melalui tiga tahapan: prepare(), bind_param(), dan execute(). Dan OWASP WSTG merekomendasikan penerapan prepared statement untuk menutup celah sql injection [10],[18].

Tabel 2. Tahapan Implementasi Prepared Statement

Tahapan	Keterangan
prepare()	Struktur query disiapkan dengan placeholder "?"; database mem-parse struktur query terlebih dahulu
bind_param()	Nilai input diikat ke placeholder sebagai data literal, bukan bagian dari sintaks SQL
execute()	Query dieksekusi; payload injeksi tidak dapat mengubah struktur query yang telah terdefinisi

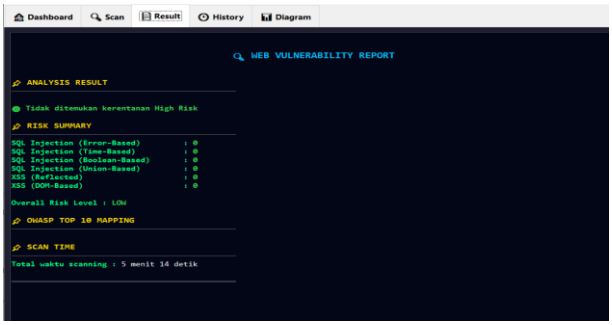
Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

4.6.2 Mitigasi XSS — Output Encoding

Mitigasi XSS dilakukan dengan *output encoding* menggunakan fungsi htmlspecialchars() dengan opsi ENT_QUOTES dan UTF-8. Fungsi ini mengonversi karakter khusus HTML menjadi entitas aman sehingga payload script tidak dieksekusi browser, melainkan ditampilkan sebagai teks biasa. OWASP WSTG merekomendasikan output encoding yang konteks-sensitif sebagai pertahanan utama terhadap seluruh varian XSS [5],[12].

4.6.3 Verifikasi Pasca-Mitigasi (Rescan)

Setelah seluruh perbaikan diterapkan, dilakukan *rescan* menggunakan scanner yang sama terhadap seluruh endpoint yang sebelumnya rentan. Tabel 3 menyajikan perbandingan status sebelum dan setelah mitigasi.



Gambar 14. Rescan Kerentanan Pada Website LisxDragon

Tabel 3. Perbandingan Status Kerentanan Sebelum dan Setelah Mitigasi

Kerentanan	Sebelum Mitigasi	Setelah Mitigasi	Status
SQLi Boolean-Based	Respon TRUE dan FALSE berbeda, menunjukkan query dapat dimanipulasi, artinya sistem menampilkan data atau informasi transaksi pengguna, Ini membuktikan input pengguna dapat dimanipulasi	Respon aplikasi konsisten, input tidak lagi memengaruhi logika query karena telah diamankan, artinya data transaksi pengguna tidak dapat dilihat lagi dengan teknik serangan sql (boolean-based)	Aman ✓
SQLi Error-Based	Muncul pesan error database, artinya Error tersebut bersifat sensitif karena menampilkan informasi internal sistem Informasi ini dapat dimanfaatkan penyerang untuk memahami struktur aplikasi dan mempermudah eksploitasi	Tidak ada pesan error yang ditampilkan, input telah divalidasi dan diamankan, artinya informasi internal sistem tidak dapat dilihat lagi dengan teknik serangan sql (error-based)	Aman ✓
SQLi Time-Based	Terjadi delay respon signifikan akibat eksekusi fungsi SLEEP, artinya terjadi gangguan layanan dalam beberapa detik	Waktu respon normal, tidak ada lagi gangguan layanan, artinya teknik serangan sql (time-based) tidak lagi berhasil	Aman ✓
SQLi Union-Based	Data tambahan muncul melalui UNION SELECT, signature unik muncul di halaman, artinya penyerang dapat mengekstrak	Signature union tidak lagi muncul, query tidak dapat dimodifikasi, hanya data yang valid yang ditampilkan, artinya teknik sql (union-	Aman ✓

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

	si data dari tabel lain dalam basis data yang seharusnya tidak ditampilkan oleh aplikasi	based) tidak lagi berhasil dan penyerang tidak dapat mengetahui lagi informasi untuk mengekstraksi data dari tabel lain	
XSS Reflected	Input pengguna ditampilkan kembali tanpa penyaringan, berpotensi menjalankan script berbahaya, artinya dapat melihat cookie pengguna, hal ini dapat membantu penyerang bisa mengambil alih sesi pengguna tanpa perlu username dan password.	Input ditampilkan dalam bentuk aman dan cookie pengguna tidak dapat dilihat lagi (tidak dieksekusi sebagai script), artinya teknik xss (reflected) tidak lagi berfungsi	Aman ✓

Hasil *rescan* mengonfirmasi bahwa seluruh 6 kerentanan yang sebelumnya teridentifikasi telah dinyatakan aman (*Not Vulnerable*). Tidak ditemukan kerentanan baru yang timbul akibat proses perbaikan. Penerapan *prepared statement* secara efektif memblokir seluruh varian SQL Injection karena database engine memproses nilai input sebagai data literal — bukan sebagai bagian dari instruksi SQL — sehingga karakter injeksi tidak dapat mengubah struktur kueri yang telah terdefinisi. Sementara itu, *output encoding* melalui htmlspecialchars() berhasil mencegah eksekusi skrip XSS karena karakter pembentuk tag script dikonversi menjadi entitas HTML yang aman sebelum ditampilkan ke browser.

Hasil penelitian ini memiliki kesamaan dengan penelitian [7], [8], dan [9] yang menunjukkan bahwa SQL Injection dan Cross-Site Scripting (XSS) masih menjadi kerentanan yang umum ditemukan pada aplikasi web. Temuan ini juga sejalan dengan penelitian [10] yang menunjukkan bahwa variasi SQL Injection tertentu memerlukan validasi lebih lanjut melalui pengujian manual untuk memperoleh hasil yang komprehensif. Selain itu, penelitian [11] menegaskan pentingnya deteksi dini terhadap SQL Injection dan XSS karena kedua jenis serangan tersebut masih menjadi ancaman utama pada aplikasi web modern. Namun, berbeda dengan penelitian sebelumnya yang umumnya berfokus pada identifikasi atau deteksi kerentanan, penelitian ini mengintegrasikan proses identifikasi, eksploitasi, mitigasi, dan verifikasi ulang dalam satu alur pengujian. Selain itu, penelitian ini tidak hanya menemukan kerentanan, tetapi juga membuktikan dampak eksploitasi melalui proof-of-concept serta memverifikasi efektivitas perbaikan melalui proses rescan yang menunjukkan bahwa seluruh kerentanan berhasil ditangani.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian ini berhasil mengidentifikasi, mengeksploitasi, dan memitigasi kerentanan keamanan pada website LisxDragon melalui metodologi *white-box penetration testing* berbasis OWASP WSTG. Dari 122 endpoint yang berhasil dipetakan melalui algoritma DFS, ditemukan 6 kerentanan valid pada 2 endpoint: SQL Injection dalam empat varian (*Error-Based, Time-Based, Boolean-Based, Union-Based*) serta *Reflected XSS*. Eksploitasi terkontrol membuktikan dampak nyata yang signifikan, mencakup kebocoran struktur basis data, kebocoran data pengguna, gangguan layanan, ekspos informasi DBMS melalui pesan kesalahan, serta pencurian cookie

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

sesi yang dapat dimanfaatkan untuk pembajakan akun.

Seluruh kerentanan berhasil dimitigasi melalui penerapan *prepared statement* untuk SQL Injection dan *output encoding* htmlspecialchars() untuk XSS — keduanya diimplementasikan langsung pada level kode sumber aplikasi. Verifikasi pasca-mitigasi melalui *rescan* mengonfirmasi bahwa seluruh endpoint yang sebelumnya rentan kini berstatus aman. Hasil ini menegaskan bahwa akar penyebab seluruh kerentanan yang ditemukan adalah ketiadaan validasi dan parameterisasi input yang konsisten. Sebagaimana ditegaskan [7],[20], Dan OWASP WSTG [12] *prepared statement* dan *output encoding* merupakan solusi yang terbukti efektif, efisien, dan mudah diimplementasikan untuk mencegah kerentanan injeksi.

5.2 Saran

(1) Pengelola platform web serupa disarankan menerapkan audit keamanan berkala yang mencakup minimal seluruh kerentanan OWASP Top 10, tidak terbatas pada SQL Injection dan XSS.

(2) Penelitian selanjutnya dapat memperluas cakupan ke kerentanan lain seperti *Stored XSS, Cross-Site Request Forgery* (CSRF), *Broken Access Control*, dan *Server-Side Request Forgery* (SSRF) untuk audit yang lebih komprehensif.

(3) Implementasi *Web Application Firewall* (WAF) direkomendasikan sebagai lapisan pertahanan tambahan untuk mendeteksi dan memblokir serangan injeksi secara real-time, melengkapi perbaikan pada level kode sumber.

(4) Pengembang aplikasi web disarankan menerapkan prinsip *secure coding* sejak fase pengembangan awal (*security by design*), sehingga kerentanan dapat dicegah sebelum sistem masuk ke tahap produksi.

6. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih yang tulus kepada Pembimbing I dan Pembimbing II atas bantuan, bimbingan, dan dukungan mereka selama proses penelitian dan penulisan artikel ini. Terima kasih juga disampaikan kepada pihak manajemen LisxDragon yang telah memberikan izin pengujian keamanan secara penuh dan sukarela, serta kepada Universitas Bumigora atas dukungan fasilitas dan suasana akademis yang kondusif. Penulis juga menyampaikan terima kasih kepada orang tua yang telah berkontribusi dan yang paling saya banggakan, yang selalu memberikan dukungan, semangat, dan doa tanpa henti sehingga artikel ini dapat diselesaikan.

DAFTAR PUSTAKA:

[1] I. Bersama, “Panduan Penanganan Insiden Serangan SQL Injection (2018) | BSSN -.” Accessed: May 18, 2026. [Online]. Available: https://ilmubersama.com/2025/04/04/panduan-penanganan-insiden-serangan-sql-injection-2018-bssn/

[2] O. Foundation, “A03 Injection - OWASP Top 10:2021.” Accessed: May 18, 2026. [Online]. Available: https://owasp.org/Top10/2021/A03_2021-Injection/index.html

[3] AnuPriya, “SQL Injection Attack Defaces Website of Indonesian Narcotics Agency,” Cyber Security News. Accessed: May 18, 2026. [Online]. Available: https://cyberpress.org/sql-injection-attack-narcotics/

[4] A. Bastian, H. Sujadi, and L. Abror, “ANALISIS KEAMANAN APLIKASI DATA POKOK PENDIDIKAN (DAPODIK) MENGGUNAKAN PENETRATION TESTING DAN SQL INJECTION,” 2020.

[5] M. Faizal Kurniawan and W. Setianto, “OPTIMASI METODE OTOMATISASI PENGHILANAGAN KERENTANAN TERHADAP SERANGAN XSS PADA APLIKASI WEB,” 2020, doi: https://doi.org/10.47775/ictech.v15i2.121.

[6] Ade Gustiyonoo, E. Irawadi Alwi, and S. Mubarak Abdullah, “Analisa Kerentanan Website Terhadap Serangan Cross-Site Scripting (XSS) Metode Penetration Testing,” *Cyber Secur. Dan Forensik Digit.*, vol. 7, no. 1, pp. 25–33, Nov. 2024, doi: 10.14421/csecurity.2024.7.1.4432.

[7] D. Ending Narhudin, B. Irawan, and A. Bahtiar, “EVALUASI KEAMANAN WEBSITE MENGGUNAKAN METODE OWASP:

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 9, Nomor 2, Juni 2026.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

PENILAIAN TERHADAP SERANGAN INJEKSI SQL DAN CROSS-SITE SCRIPTING (XSS)," *JATI J. Mhs. Tek. Inform.*, vol. 8, no. 1, pp. 675–680, Feb. 2024, doi: 10.36040/jati.v8i1.8700.

[8] T. Anugrah, "PENETRATION TESTING KEAMANAN WEBSITE STIE SAMARINDA MENGGUNAKAN TEKNIK SQL INJECTION DAN XSS," *J. Inform. Dan Tek. Elektro Terap.*, vol. 12, no. 1, Jan. 2024, doi: 10.23960/jitet.v12i1.3882.

[9] Y. Sitorus, "ANALISIS KEAMANAN WEBSITE XYZ TERHADAP SERANGAN SQL INJECTION DAN CROSS SITE SCRIPTING (XSS)," vol. 7, no. 2, 2025.

[10] R. Fernanda, M. Data, and F. A. Bakhtiar, "ANALISIS EFEKTIVITAS OWASP ZAP DALAM MENDETEKSI KERENTANAN TERHADAP SERANGAN SQL INJECTION".

[11] R. Bakır, "UniEmbed: A Novel Approach to Detect XSS and SQL Injection Attacks Leveraging Multiple Feature Fusion with Machine Learning Techniques," *Arab. J. Sci. Eng.*, vol. 50, no. 19, pp. 15591–15604, Oct. 2025, doi: 10.1007/s13369-024-09916-4.

[12] O. Foundation, "OWASP Web Security Testing Guide | OWASP Foundation." Accessed: May 18, 2026. [Online]. Available: https://owasp.org/www-project-web-security-testing-guide/

[13] O. Pribadi, "Maze Generator Dengan Menggunakan Algoritma Depth-First-Search," *J. TIMES*, vol. 4, no. 1, pp. 1–5, Jul. 2015, doi: 10.51351/jtm.4.1.2015213.

[14] S. Fernandez, A. W. Mahfuzhi, and B. Rabbani, "Pemanfaatan Algoritma Sequential Search Dalam Penerapan Aplikasi Inventaris," *J. Electr. Eng. Comput. JEECOM*, vol. 6, no. 1, pp. 58–67, May 2024, doi: 10.33650/jeecom.v6i1.8371.

[15] M. N. Fakhrizal, "Penerapan Algoritma Breadth-First Search dan Exhaustive Search dalam Menentukan Lokasi Titik Kumpul dalam Ruang Publik".

[16] S. Lika, R. D. P. Halim, and I. Verdian, "ANALISA SERANGAN SQL INJEKSI MENGGUNAKAN SQLMAP," vol. 4, 2018.

[17] N. Augusta, A. I. Hadiana, and F. R. Umbara, "Sistem Keamanan Website Dengan Multi Metode Untuk Mencegah SQL Injection," 2024.

[18] A. N. Maulana, M. Data, and F. A. Bakhtiar, "Perancangan dan Implementasi Snort Rule Set untuk Deteksi Serangan SQL Injection," 2025.

[19] I. Laleb, "ANALISIS CROSS-SITE SCRIPTING (XSS) INJECTION – REFLECTED XSS AND STORED XSS MENGGGUNAKAN FRAMEWORK OWASP 10," *J. Ilm. Flash*, vol. 8, no. 1, p. 36, Jan. 2023, doi: 10.32511/flash.v8i1.952.

[20] W. F. Enjelina and A. Tedyyana, "IMPLEMENTASI PREPARED STATEMENT PADA FORM INPUT UNTUK MENCEGAH SQL INJECTION PADA APLIKASI WEB PEMINJAMAN FASILITAS," vol. 10, no. 1, 2026.