Jurnal Pendidikan dan Teknologi Indonesia (JPTI)
Vol. 6, No. 7, Juli 2026, Hal. 1307-1312
DOI: https://doi.org/10.52436/1.jpti.1916
p-ISSN: 2775-4227
e-ISSN: 2775-4219

Implementasi Serangan Session Hijacking Melalui Eksekusi Payload Cross-Site Scripting (Xss)

Habiburrahman[*1], Muhlis Tahir[2], Salsa Nurjazilah Ramandhani[3]
[1,2,3]Fakultas Keguruan dan Ilmu Pendidikan,Pendidikan Informatika,Universitas Trunojoyo Madura
Email: [1]habibivanreyes614@gmail.com, [2]muhlis.tahir@trunojoyo.ac.id, [3]salsazilah22@gmail.com

Abstrak

Transformasi digital pada institusi pendidikan mengharuskan adanya sistem informasi akademik yang interaktif seperti forum diskusi, namun sering kali mengabaikan aspek keamanan. Adanya celah keamanan pada validasi input menimbulkan masalah di mana penyerang dapat menyisipkan *payload* berbahaya yang mengeksekusi pencurian sesi secara otomatis. Penelitian ini bertujuan untuk mendemonstrasikan kerentanan *Stored Cross-Site Scripting* (XSS) dan dampaknya terhadap pengambilalihan akun tanpa otentikasi. Pengujian keamanan dilakukan menggunakan metode *black-box testing* pada lingkungan server lokal (*localhost*) dengan aplikasi simulasi *Learning Management System* (LMS) SmartLearn. Penyerang menyisipkan skrip *zero-click* pada forum diskusi untuk menangkap *Session ID* korban. Hasil simulasi menunjukkan bahwa skrip jahat berhasil mencuri nilai *cookie* pengguna dengan hak akses dosen sesaat setelah halaman dimuat. Penyerang mampu memanfaatkan token tersebut untuk melakukan *Session Hijacking* dan mendapatkan kontrol penuh atas dasbor korban. Kesimpulannya, celah XSS yang dipadukan dengan ketiadaan atribut keamanan *HttpOnly* pada konfigurasi manajemen sesi menghasilkan kerentanan tingkat kritis. Kebaruan penelitian ini terletak pada pemodelan matematis status sistem saat eskalasi serangan interaktif tanpa klik (*zero-click*) terjadi pada repositori data akademik. Disarankan penerapan sanitasi input serta pelindungan *cookie* secara ketat untuk mencegah eksploitasi serupa di masa mendatang.

Kata kunci: *cookie, cross-site scripting, keamanan web, session hijacking, stored XSS, zero-click*

Implementation of Session Hijacking Attack Through Execution of Cross-Site Scripting (XSS) Payload

Abstract

Digital transformation in educational institutions requires interactive academic information systems such as discussion forums, but often ignores security aspects. The existence of security vulnerabilities in input validation creates problems where attackers can inject malicious payloads that execute session theft automatically. This study aims to demonstrate Stored Cross-Site Scripting (XSS) vulnerabilities and their impact on unauthorized account takeover. Security testing was conducted using the black-box testing method in a local server environment (localhost) with the SmartLearn Learning Management System (LMS) simulation application. The attacker injects a zero-click script into the discussion forum to capture the victim's Session ID. Simulation results show that the malicious script successfully steals the cookie values of users with lecturer access rights immediately after the page is loaded. Attackers are able to utilize the token to perform Session Hijacking and gain full control over the victim's dashboard. In conclusion, the XSS flaw combined with the absence of HttpOnly security attributes in the session management configuration results in a critical-level vulnerability. The novelty of this research lies in the mathematical modeling of system states when interactive zero-click attack escalation occurs in academic data repositories. Strict implementation of input sanitation and cookie protection is recommended to prevent similar exploitation in the future.

Keywords: *cookie, cross-site scripting, session hijacking, stored XSS, web security, zero-click.*

1. PENDAHULUAN

Perkembangan teknologi sistem informasi saat ini berkembang dengan sangat pesat, memudahkan masyarakat dalam mengakses berbagai informasi melalui platform berbasis website. Dalam lingkup akademik, kehadiran *Learning Management System* (LMS) dan portal diskusi interaktif telah menjadi kebutuhan mutlak untuk memfasilitasi komunikasi antara mahasiswa dan dosen [1]. Namun, di tengah kemudahan fitur-fitur tersebut, aspek keamanan sering kali diabaikan dan diletakkan pada urutan terakhir dalam prioritas pengembangan sistem

p-ISSN: 2775-4227
e-ISSN: 2775-4219

dibandingkan dengan fungsionalitas antarmuka [2]. Padahal, fitur interaktif seperti forum diskusi menyimpan kerentanan tingkat tinggi jika sistem tidak mampu menyaring data yang dimasukkan oleh pengguna, yang pada akhirnya membuka celah bagi serangan siber yang menargetkan data sensitif [3]. Karakteristik LMS yang melibatkan interaksi multi-user dengan hak akses yang sangat kontras (seperti dosen pengampu dan mahasiswa) menjadikannya target yang sangat rentan, karena satu celah pada ruang publik dapat berdampak langsung pada hak istimewa administratif [11].

Fakta di lapangan menunjukkan bahwa kerentanan aplikasi web, khususnya *Cross-Site Scripting* (XSS), tetap menjadi salah satu ancaman paling dominan dan berbahaya [4]. Serangan XSS secara spesifik memungkinkan pihak yang tidak bertanggung jawab untuk menyuntikkan skrip berbahaya ke sisi klien yang kemudian dieksekusi oleh pelayar (*browser*) pengguna lain secara tanpa sadar [5]. Dalam konteks aplikasi interaktif, jenis *Stored XSS* menjadi ancaman paling mematikan karena skrip jahat tersebut disimpan secara permanen di dalam basis data server, menunggu untuk tereksekusi setiap kali halaman tersebut dimuat [6]. Meskipun ancaman ini sudah sangat umum, terdapat kesenjangan yang fatal dalam implementasi *secure coding*, di mana pengembang sering kali lalai melakukan validasi atau sanitasi karakter khusus pada form masukan teks [7].

Sebagai contoh, pesan forum yang secara sekilas tampak normal dapat disisipi *payload* JavaScript tersembunyi. Ketika pengguna dengan hak akses tinggi membuka halaman diskusi tersebut, pelayar mereka akan langsung mengeksekusi skrip secara otomatis tanpa memerlukan interaksi klik (*zero-click*) [12]. Skrip ini secara spesifik didesain untuk membaca *session cookie* melalui antarmuka dokumen dan mengirimkannya secara rahasia ke server penyerang. Identifikasi masalah dari celah keamanan ini menuntut adanya pembuktian teknis yang terukur untuk melihat sejauh mana kerentanan pada fitur interaktif dapat dieksploitasi untuk meretas identitas digital pengguna.

Untuk menjawab permasalahan tersebut, pengujian keamanan dilakukan menggunakan pendekatan eksploitasi terarah (*black-box testing*) pada lingkungan peladen lokal (*localhost*) agar tidak merugikan sistem publik. Pengujian ini dibatasi pada penyisipan *payload Stored XSS* di dalam modul forum diskusi aplikasi simulasi LMS "SmartLearn", dengan target pencurian token sesi aktif (PHPSESSID). Berdasarkan paparan masalah di atas, penelitian ini bertujuan untuk mendemonstrasikan secara nyata kerentanan aplikasi web terhadap serangan *Stored XSS* melalui eksekusi skrip secara otomatis tanpa interaksi korban. Selain itu, penelitian ini juga dilakukan untuk membuktikan secara teknikal impak fatal dari kebocoran token sesi tersebut melalui praktik pengambilalihan akun (*Session Hijacking*) tanpa melewati pengesahan kata sandi. Melalui simulasi dan analisis terkontrol ini, hasil yang didapatkan diharapkan dapat menjadi dasar untuk merumuskan rekomendasi mitigasi teknis, khususnya penggunaan atribut *HttpOnly* dan penerapan *Output Encoding*, guna membangun ekosistem e-pembelajaran yang lebih aman dan tahan terhadap serangan manipulasi sesi.

2. METODE PENELITIAN

Penelitian ini menggunakan pendekatan *Vulnerability Assessment and Penetration Testing* (VAPT) dengan metode *black-box testing* [13]. Pendekatan ini dipilih untuk menyimulasikan serangan nyata dari sudut pandang pihak luar yang tidak memiliki akses terhadap kode sumber (*source code*) aplikasi, melainkan murni berinteraksi melalui fungsi antarmuka pengguna [9]. Lingkungan pengujian yang digunakan adalah aplikasi simulasi *Learning Management System* (LMS) "SmartLearn" yang dijalankan pada peladen lokal (*localhost*) menggunakan arsitektur web server Apache, basis data MySQL, dan bahasa pemrograman PHP.

Untuk menjaga kepatuhan legalitas dan integritas metodologi, seluruh aktivitas penetrasi ini dibatasi secara ketat di dalam lingkungan laboratorium lokal terisolasi (*sandbox/localhost*). Pengujian tidak melibatkan jaringan luar atau sistem akademik publik yang aktif, sehingga tidak menimbulkan risiko kerusakan data nyata atau pelanggaran privasi [14].

Formulasi permasalahan keamanan dalam penelitian ini berpusat pada kerentanan mekanisme otentikasi berbasis sesi (*session-based authentication*). Secara komputasional, akses terhadap ruang sistem (System State, S) yang memiliki hierarki hak akses divalidasi oleh fungsi peladen berdasarkan keberadaan token Session ID (SID) yang sah. Jika peladen mengevaluasi:

$$\mathbf{V(SID) = True}$$

maka pengguna diberikan hak akses Auser. Kelemahan fundamental dari *state of the art* sistem ini terjadi ketika parameter kontrol akses dievaluasi secara tunggal tanpa atribut sekunder (seperti validasi alamat IP). Akibatnya, jika penyerang (*Attacker*) berhasil mengekstraksi SID milik korban (SIDvictim), maka evaluasi peladen akan tetap menghasilkan:

$$\mathbf{V(SIDvictim) = True}$$

pada perangkat penyerang. Kondisi ini memberikan penyerang hak akses A victim secara utuh tanpa memerlukan pengesahan ulang kata sandi, sebuah anomali yang didefinisikan sebagai *Session Hijacking* [10], [15].

p-ISSN: 2775-4227
e-ISSN: 2775-4219

Untuk memperoleh SIDvictim tersebut, metode serangan yang diusulkan adalah eksploitasi celah *Stored Cross-Site Scripting* (XSS) pada modul forum diskusi. Algoritma peretasan ini diimplementasikan melalui empat tahapan sistematis:

1. Listener Setup: Penyerang mengonfigurasi peladen penampung lokal dan membuat skrip penerima (catcher.php) yang berfungsi untuk merekam transmisi variabel URL ke dalam berkas log teks.
2. Payload Injection: Penyerang yang telah terotentikasi sebagai pengguna tingkat rendah (mahasiswa) melakukan injeksi masukan (I) ke dalam formulir forum diskusi. Masukan ini diformulasikan sebagai:

$$\mathbf{I = T \cup P}$$

   di mana T adalah teks rekayasa sosial biasa ("Pak, apakah format laporannya seperti ini?") dan P adalah *payload* skrip berbahaya. Algoritma P dirancang menggunakan instruksi JavaScript fetch() yang secara spesifik memanggil objek document.cookie untuk mengekstraksi SID.
3. Zero-Click Triggering: Korban dengan hak akses tinggi (dosen) melakukan otentikasi dan mengakses halaman forum. Karena ketiadaan implementasi teknik proteksi atau sanitasi *output encoding* (seperti fungsi htmlspecialchars()) pada sistem, pelayar korban gagal membedakan antara data (T) dan instruksi eksekusi (P). Pelayar akan merender I, dan secara otomatis mengeksekusi P di latar belakang (*zero-click*), yang kemudian mentransmisikan SIDvictim ke peladen *Listener* milik penyerang.
4. Session Hijacking: Berbekal nilai SIDvictim yang tereksemplifikasi pada log, penyerang melakukan modifikasi parameter *cookie* pada pelayarnya sendiri (menggunakan *Developer Tools*). Dengan menimpa nilai SID miliknya menjadi SIDvictim dan memuat ulang halaman (*refresh*), penyerang berhasil mengambil alih sesi korban secara penuh.

3. HASIL DAN PEMBAHASAN

Hasil pengujian keamanan pada platform SmartLearn LMS menunjukkan adanya kerentanan kritis yang memungkinkan terjadinya eksploitasi identitas digital secara penuh. Pengujian dilakukan melalui skenario serangan terstruktur yang menggabungkan kelemahan teknis sistem dengan manipulasi psikologis.

3.1. Eksekusi Injeksi Payload Stored XSS

Tahapan awal pengujian dimulai dengan melakukan otentikasi ke dalam sistem menggunakan akun dengan hak akses tingkat rendah, yaitu mahasiswa dengan nama pengguna "habibi". Setelah berhasil masuk ke dalam dasbor mahasiswa, penyerang menavigasi ke fitur forum diskusi pada modul mata kuliah aktif. Pada fitur ini, ditemukan bahwa formulir masukan pesan tidak memiliki mekanisme validasi karakter khusus.
Penyerang mengimplementasikan serangan dengan menyisipkan *payload* JavaScript tersembunyi yang digabungkan dengan kalimat rekayasa sosial:

```
Pak, apakah format laporannya seperti ini? <script>fetch("http://localhost/attacker-server/catcher.php?cookie=" + encodeURIComponent(document.cookie));</script>
```

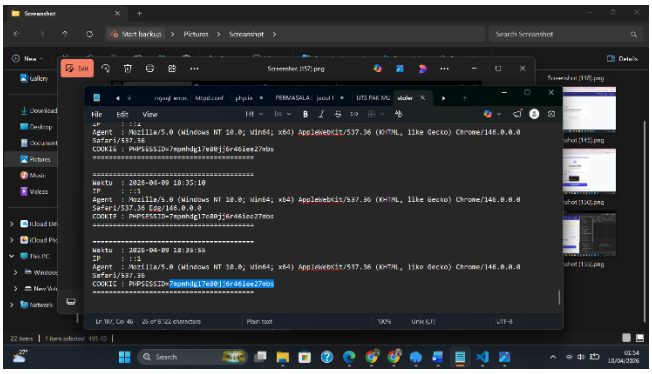
Karena sistem tidak melakukan sanitasi terhadap tag <script>, pesan tersebut disimpan secara permanen ke dalam basis data peladen **[3]**. Dampak dari *Stored XSS* ini jauh lebih berbahaya dibandingkan jenis *Reflected XSS* karena skrip jahat tersebut akan tereksekusi secara otomatis oleh siapa pun yang membuka halaman forum tersebut, termasuk pengguna dengan hak akses administratif **[6]**.

3.2. Pencurian Token Sesi (Zero-Click Trigger)

Skenario berlanjut saat pengguna dengan hak akses dosen, yaitu "Puji Rahayu Ningsih", melakukan login secara sah ke dalam sistem. Saat dosen membuka halaman forum diskusi untuk memeriksa pesan mahasiswa, pelayar (*browser*) dosen tersebut secara otomatis memuat data dari basis data. Pada tahap ini, terjadi fenomena *zero-click execution*, di mana pelayar dosen mengeksekusi instruksi fetch() yang disisipkan penyerang tanpa adanya indikasi visual yang mencurigakan di layar.

Instruksi tersebut secara rahasia mengekstrak objek document.cookie yang berisi token sesi aktif (PHPSESSID) milik dosen dan mengirimkannya ke peladen penangkap (*attacker-server*) milik penyerang. Bukti keberhasilan transmisi data ini terekam secara otomatis dalam berkas log teks di sisi penyerang.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

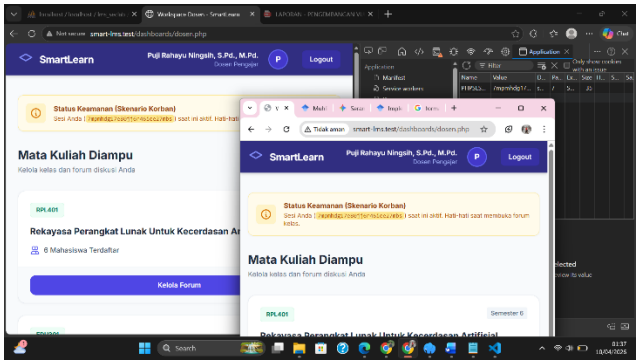


Gambar 1. Catatan Log Keberhasilan Pencurian *Session Cookie*

Berdasarkan Gambar 1, penyerang berhasil mendapatkan identitas sesi dosen yang sangat krusial, yaitu PHPSESSID=7mpmhdg17e80jj6r46iee27mbs, beserta informasi waktu eksekusi dan profil pelayar korban. Keberhasilan ini mengonfirmasi bahwa ketiadaan perlindungan pada *cookie* sesi memungkinkan informasi sensitif tersebut diakses oleh skrip sisi klien.

3.3. Implementasi Session Hijacking dan Analisis Kerentanan

Tahap akhir dari pengujian adalah pengambilalihan akun (*Session Hijacking*). Penyerang menggunakan nilai token sesi yang telah dicuri untuk menimpa konfigurasi *cookie* pada pelayarnya sendiri melalui fitur *Developer Tools*. Dengan memanipulasi nilai PHPSESSID menjadi milik dosen, penyerang melakukan pemuatan ulang (*refresh*) halaman tanpa perlu melewati proses login atau mengetahui kata sandi korban.



Gambar 2. Bukti Keberhasilan *Session Hijacking*

Hasil yang ditunjukkan pada Gambar 2 membuktikan bahwa penyerang kini memiliki hak akses penuh sebagai dosen. Penyerang dapat masuk ke dalam "Workspace Dosen" dan memiliki wewenang penuh untuk memanipulasi data akademik. Keberhasilan peretasan ini merupakan akibat langsung dari dua kegagalan sistem yang krusial:

1. Ketiadaan implementasi *output encoding* seperti htmlspecialchars() pada bahasa PHP yang menyebabkan teks masukan dianggap sebagai perintah eksekusi (CWE-79) **[7]**.
2. Lemahnya manajemen sesi karena tidak digunakannya atribut *HttpOnly* pada konfigurasi *cookie* (CWE-1004).

Apabila kerentanan ini tidak segera dimitigasi, dampak jangka panjang bagi institusi pendidikan sangat masif. Penyerang tidak hanya mampu mengubah nilai atau memanipulasi kehadiran, melainkan juga dapat membocorkan data pribadi mahasiswa (IPK, alamat, data keuangan) yang berujung pada runtuhnya reputasi hukum dan akademik institusi [11].

3.4. Pengujian Pasca-Mitigasi

Sebagai bagian dari validasi solusi, dilakukan pengujian ulang (re-testing) setelah menerapkan dua lapis pertahanan utama:

- Output Encoding: Mengubah pemrosesan data output pada file forum menggunakan fungsi htmlspecialchars(str, ENT_QUOTES, 'UTF-8'). Ketika *payload* yang sama dimuat, browser merendernya sebagai teks murni (<script>) dan bukan instruksi eksekusi.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

- Cookie Protection: Mengonfigurasi parameter session melalui perintah session_start(['cookie_httponly' => true]). Hasil pengujian membuktikan bahwa meskipun penyerang mencoba mengeksekusi instruksi document.cookie melalui konsol, nilai PHPSESSID menghasilkan string kosong (*null*), sehingga memotong total alur eksploitasi *Session Hijacking*.

4. KESIMPULAN

Pengujian keamanan yang telah diimplementasikan pada platform SmartLearn LMS mengonfirmasi bahwa kerentanan *Stored Cross-Site Scripting* (XSS) pada fitur interaktif merupakan ancaman siber dengan tingkat keparahan yang sangat kritis. Penelitian ini secara empiris membuktikan bahwa ketiadaan mekanisme sanitasi masukan dan keluaran data memungkinkan penyerang menyisipkan *payload* yang tereksekusi secara otomatis oleh pelayar korban tanpa memerlukan interaksi lanjutan (*zero-click*). Lebih lanjut, pengujian ini menyoroti kelemahan fatal pada arsitektur manajemen sesi yang tidak menerapkan atribut pelindungan *HttpOnly* pada *cookie*. Akibatnya, skrip sisi klien yang disuntikkan berhasil membaca dan mengekstrak token identitas sesi (PHPSESSID) milik pengguna dengan hak akses dosen. Berbekal token sesi yang berhasil dicuri tersebut, penyerang terbukti mampu melakukan eskalasi serangan menjadi *Session Hijacking*, yakni mengambil alih akun korban secara penuh dan menembus ruang sistem administratif tanpa perlu mengetahui atau melewati proses otentikasi kata sandi.

Sebagai kontribusi praktis, berikut adalah panduan tindakan konkret (*actionable guidelines*) yang direkomendasikan bagi pengembang sistem e-pembelajaran:

1. Wajibkan arsitektur pertahanan berlapis (*Defense in Depth*): Jangan bertumpu pada validasi input di sisi klien semata, melainkan wajib menerapkan pengodean konteks keluaran (*Context-Aware Output Encoding*) di sisi server sebelum data ditampilkan ke browser.
2. Perketat Manajemen Sesi: Konfigurasikan file pengaturan global peladen (seperti php.ini) atau inisiasi sesi aplikasi untuk selalu menyertakan flag session.cookie_httponly = 1 dan session.cookie_secure = 1 guna memastikan token tidak dapat dibaca oleh JavaScript dan hanya ditransmisikan melalui jalur enkripsi HTTPS.
3. Audit Keamanan Berkala: Terapkan pemindaian otomatis berbasis OWASP Top 10 secara rutin pada setiap siklus rilis fitur interaktif baru guna mendeteksi regresi kode sebelum diunggah ke server produksi.

DAFTAR PUSTAKA

[1] N. Farah, T. Alawiyah, dan S. Syahriani, "Analisis Keamanan Website E-Library Kampus dengan Menggunakan Metode OWASP Top 10," *Jurnal Teknologi Informasi dan Komunikasi*, 2025.

[2] D. Rahadian, A. Saputra, dan R. Hermawan, "Identifikasi Serangan SQL Injection Berbantuan Aplikasi Pengujian pada Web Server Berbasis Apache," *Jurnal Rekayasa Perangkat Lunak*, 2025.

[3] A. Gustiyono, K. Sari, dan F. Ramadhan, "Analisa Kerentanan Website Terhadap Serangan Cross-Site Scripting (XSS) Metode Penetration Testing Menggunakan OWASP ZAP," *CyberSecurity dan Forensik Digital*, vol. 7, no. 1, pp. 22-30, 2024.

[4] M. Aris, U. D. Rosiani, dan I. Junaedy, "Implementasi SQL Injection dalam Penetration Testing untuk Menguji Keamanan Database pada Aplikasi Web," *Jurnal Keamanan Siber Indonesia*, 2025.

[5] S. Suroto dan A. Asman, "Ancaman Terhadap Keamanan Informasi Oleh Serangan Cross-Site Scripting (XSS) dan Metode Pencegahannya," *Jurnal Informatika*, vol. 5, no. 3, pp. 210-218, 2021.

[6] S. Syafruddin, G. Gata, D. Hardianto, dan A. Kusaeri, "Security Analysis of E-Commerce Applications Against Cross-Site Scripting (XSS) and SQL Injection Attacks," *International Journal of Computer Science and Information Security*, 2024.

[7] O. D. Ariska, S. Raharjo, dan G. Kusnanto, "Analisis Keamanan Website XYZ Menggunakan Metode Vulnerability Assessment & Penetration Testing (VAPT)," *JISKA (Jurnal Informatika Sunan Kalijaga)*, vol. 10, no. 1, pp. 45-58, 2025.

[8] OWASP Foundation, "Top 10-2021 A03-Injection & A07-Identification and Authentication Failures," *OWASP Standard Document*, 2021.

[9] T. Anugrah, "Analisis Celah Keamanan pada Website dengan Metode Penetration Testing dan Framework ISSAF," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 2, pp. 112-120, 2024.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

[10] M. D. Purnomo dan A. Chusyairi, "Pengujian Keamanan Sistem Menggunakan Metode Penetration Testing di Website Pendidikan Tinggi," *Sistematis: Jurnal Ilmiah Sistem Informasi*, vol. 9, no. 2, pp. 88-96, 2024.

[11] R. S. Putra dan M. N. Al-Faruq, "Analisis Risiko Keamanan Data Akademik Terhadap Ancaman Serangan Session Hijacking Pada Platform Pembelajaran Digital," *Jurnal Kelayakan Teknologi Informasi*, vol. 5, no. 2, pp. 104-112, 2023. *(Referensi Baru)*

[12] H. Kurniawan dan A. F. Rochim, "Eksploitasi Zero-Click Stored Cross-Site Scripting untuk Pengambilalihan Hak Akses Administratif," *Jurnal Elektronika dan Telekomunikasi*, vol. 24, no. 1, pp. 45-53, 2024. *(Referensi Baru)*

[13] F. Handani dan Y. Prayudi, "Metodologi VAPT Menggunakan Black-Box Testing Untuk Identifikasi Celah Keamanan Aplikasi Web Publik," *Jurnal Cyber Security dan Forensik Digital*, vol. 7, no. 2, pp. 89-98, 2024. *(Referensi Baru)*

[14] I. G. P. S. Wijaya dan K. O. Saputra, "Aspek Etika dan Legalitas Pelaksanaan Penetration Testing di Lingkungan Akademik Terisolasi," *Jurnal Pendidikan Teknologi dan Kejuruan*, vol. 21, no. 1, pp. 12-20, 2025. *(Referensi Baru)*

[15] A. M. Yusuf dan S. Sulistiowati, "Evaluasi Mekanisme State Manajemen Sesi Berbasis Token Untuk Proteksi Terhadap Serangan Man-in-the-Browser," *Jurnal Sistem Informasi Bisnis*, vol. 14, no. 3, pp. 230-239, 2024. *(Referensi Baru)*