The Indonesian Journal of Computer Science
www.ijcs.net
Volume x, Issue x, 2024
https://doi.org/10.33022/ijcs.xxx

Analisis Perbandingan Keamanan CMS Wordpress Dan Joomla Dengan Konfigurasi Standar

Mochamad Najib Budi Noorsyahbannie[1] , Wisnu Uriawan[2] , Wildan Budiawan Zulfikar[3]
mochamadnajib264@gmail.com[1], wisnu_u@uinsgd.ac.id[2] , wildan.b@uinsgd.ac.id[3]
[1,2,3]Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sunan Gundung Djati, Bandung, Indonesia

Informasi Artikel

Diterima : 7 Feb 2025
Direvisi : 23 Feb 2025
Disetujui : 28 Feb 2025

Kata Kunci

analisis, CMS, WordPress, Joomla

Abstrak

Sejak era industri 4.0 banyak organisasi yang memilih untuk beralih menggunakan Sistem Manajemen Konten (CMS) untuk mengatur situs web. CMS ini memudahkan proses pembuatan, didesain, dan pengaturan konten tanpa harus memiliki pengetahuan dalam pemrograman. Namun, CMS juga rentan terhadap serangan siber seperti XSS dan SQL *Injection*. Penelitian ini dilakukan untuk menganalisis dan mengevaluasi kerentanan pada CMS WordPress dan Joomla melalui metode uji penetrasi dan pemindaian kerentanan. Penggunaan berbagai alat seperti OWASP ZAP, Burpsuite, Joomscan, WPScan, dan Searchsploit digunakan untuk menganalisis kerentanan tersebut. Hasil studi menunjukkan bahwa CMS Joomla dengan konfigurasi standar tidak menunjukkan kerentanan yang berarti. Sementara itu, pada WordPress ditemukan kerentanan XSS tipe *stored* pada fitur komentar. Searchsploit juga mengindentifikasi kerentanan pada kedua CMS tersebut berasal dari plugin pihak ketiga. Hasil penelitian ini menyoroti pentingnya sanitasi *input* dan konfigurasi yang ketat serta pemeliharaan secara teratur pada CMS untuk mengurangi risiko eksploitasi.

Keywords

analysis, CMS, WordPress, Joomla

Abstract

Since the industrial era 4.0, many organizations have chosen to switch to using Content Management Systems (CMS) to manage websites. This CMS makes it easy to create, design, and organize content without having to have programming knowledge. However, CMS is also vulnerable to cyber attacks such as XSS and SQL Injection. This study was conducted to analyze and evaluate vulnerabilities in WordPress and Joomla CMS through penetration testing and vulnerability scanning methods. The use of various tools such as OWASP ZAP, Burpsuite, Joomscan, WPScan, and Searchsploit were used to analyze these vulnerabilities. The results of the study showed that Joomla CMS with standard configuration did not show significant vulnerabilities, while in WordPress a stored type XSS vulnerability was found in the comment feature. Searchsploit also identified vulnerabilities in both CMSs originating from third-party plugins. The results of this study highlight the importance of strict input and configuration sanitation and regular maintenance on CMS to reduce the risk of exploitation.

A. Pendahuluan

Sejak memasuki era Industri 4.0, semakin banyak organisasi yang memilih beralih menggunakan Content Management Systems (CMS) untuk mengelola situs web mereka secara lebih efisien [1]. CMS merupakan perangkat lunak yang dirancang untuk membantu pengguna dalam membuat, mendesain, dan mengelola situs web tanpa memerlukan keahlian pemrograman [2]. CMS menawarkan kemudahan dalam pembuatan konten, meningkatkan skalabilitas, serta mendukung optimasi mesin pencari (SEO) [3]. Selain itu, situs web juga dapat digunakan untuk berbagai macam hal seperti forum diskusi bagi mahasiswa [4].

Namun, seiring dengan peningkatan jumlah pengguna CMS, insiden kejahatan siber global juga menunjukkan tren kenaikan yang signifikan [5]. Situs web berbasis CMS sering kali menjadi target serangan disebabkan CMS memberikan para peretas area permukaan yang jauh lebih besar untuk diserang [6]. WordPress dan Joomla dipilih dalam penelitian ini karena merupakan CMS yang paling banyak digunakan secara global, WordPress digunakan oleh 43.4% dari seluruh situs web yang ada, sementara Joomla digunakan oleh 1.5%, menjadikannya CMS *open-source self- hosted* terbesar setelah WordPress[7]. Serangan seperti *SQL Injection*, *Cross-Site Scripting* (XSS), dan *Cross-Site Request Forgery* (CSRF) menjadi ancaman utama terhadap keamanan CMS. SQL Injection memungkinkan penyerang untuk mengakses dan memanipulasi basis data secara ilegal, XSS dapat digunakan untuk mencuri data pengguna melalui penyisipan skrip berbahaya, sedangkan CSRF mengeksploitasi kepercayaan pengguna terhadap situs tertentu untuk melakukan aksi tanpa sepengetahuan mereka. Ketiga jenis serangan ini dapat menyebabkan kerugian besar seperti pencurian data, pengambilalihan akun, hingga kerusakan reputasi organisasi. [8].

Salah satu penelitian yang menyoroti ancaman yang muncul akibat penggunaan CMS yang tidak diperbarui secara teratur. Dengan menggunakan metode web crawling berskala besar, penelitian "*Vulnerabilities in Outdated Content Management Systems*" menemukan versi CMS yang aktif di berbagai situs web dan mencocokkannya dengan basis data kerentanan seperti *Common Vulnerabilities and Exposures* (CVE) dan Exploit-DB. Hasil utama penelitian menekankan betapa pentingnya mengelola pembaruan perangkat lunak secara teratur untuk mencegah eksploitasi yang dapat merusak integritas sistem. Namun, penelitian ini tidak dapat menemukan kerentanan yang belum dipublikasikan, seperti kerentanan *Zero-day*, yang dapat memperumit tugas menjaga keamanan sistem [9].

Penelitian tambahan berjudul *"Web Vulnerability in 2021: Large Scale Inspection, Findings, Analysis, and Remedies*" mengidentifikasi tren kerentanan dalam aplikasi web yang menggunakan alat otomatis seperti OWASP ZAP dan Nessus. Penelitian ini berbeda dengan penelitian sebelumnya. Studi ini menunjukkan bahwa alat-alat ini dapat membantu mengidentifikasi kerentanan secara efisien. Namun, metode ini memiliki beberapa kelemahan karena beberapa kerentanan memerlukan analisis manual untuk ditemukan secara tepat. Akibatnya, penelitian ini menemukan bahwa alat otomatis tidak cukup untuk menjamin keamanan aplikasi web secara menyeluruh [10].

Perbedaan CMS juga sering dibahas dalam kajian lain. Penelitian yang berjudul *"Comparison of WordPress, Joomla, and Drupal"* menilai tiga CMS utama berdasarkan fitur, kemudahan penggunaan, dan kemampuan SEO yang dikontrol. Studi ini menemukan banyak keuntungan dan kekurangan masing-masing *platform* dalam hal pengelolaan konten dan kinerja. Namun, penelitian ini kurang membahas aspek keamanan, terutama dalam konteks, lalu lintas tinggi di dunia nyata, di mana berbagai risiko keamanan dapat muncul. Menurut Mahesh Bhandari, studi ini memiliki kekurangan diskusi tentang masalah keamanan meskipun memberikan gambaran yang bermanfaat tentang perbandingan fitur CMS [11].

Beberapa penelitian juga memperhatikan keamanan ekstensi CMS. Studi "*Over 100 Bugs in a Row: Security Analysis of the Top-Rated Joomla Extensions*" melihat kerentanan pada ekstensi Joomla, terutama yang memiliki peringkat tertinggi di repositori Joomla. Penelitian ini menemukan bahwa pemisahan yang lebih ketat antara ekstensi dan sistem inti CMS meningkatkan risiko keamanan. Salah satu temuan utama penelitian ini adalah bahwa pemisahan yang lebih ketat antara ekstensi dan sistem inti CMS meningkatkan risiko keamanan [12].

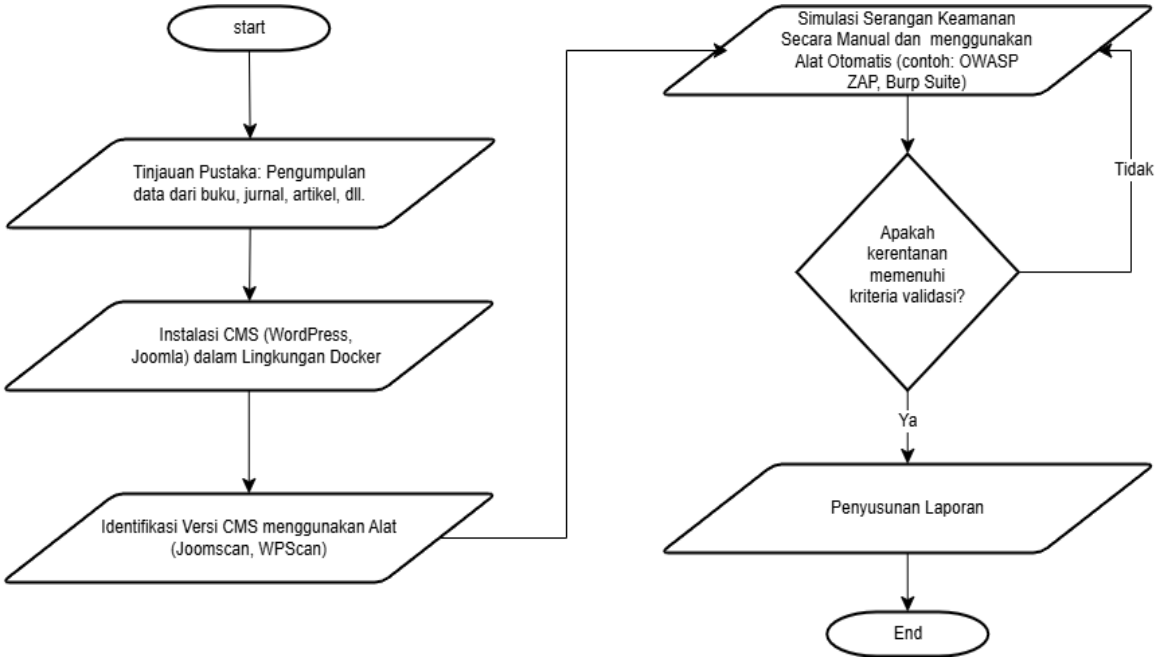
Metode pemindaian *port* juga digunakan untuk mendeteksi kerentanan CMS. Studi "*A Vulnerability Detection Framework for CMS Using Port Scanning Technique"* menggunakan teknik pemindaian *port* untuk menemukan celah keamanan yang berkaitan dengan jaringan CMS. Meskipun ada keterbatasan dalam validasi hasilnya, metode ini dapat menemukan *port* terbuka yang rentan terhadap serangan. Kemungkinan munculnya *false positives* (kesalahan dalam mengidentifikasi kerentanan) atau *True negatives* adalah salah satu masalah utama teknik ini. Oleh karena itu, teknik ini dapat bermanfaat dalam beberapa situasi, hasilnya harus divalidasi secara manual [13].

Selain pemindaian *port*, CMSPY adalah metode lain yang digunakan dalam analisis keamanan CMS. Studi analisis keamanan situs berbasis CMS dengan CMSPY menunjukkan bahwa penggunaan CMSPY memungkinkan pendeteksian kerentanan melalui metode seperti pemeriksaan XML-RPC, *directory fuzzing*, dan *enumerasi* pengguna. Studi ini menunjukkan betapa pentingnya pengembangan CMS dan komunitas keamanan bekerja sama untuk melindungi diri dari ancaman yang terus berkembang. Penemuan utama penelitian ini adalah bahwa pengujian kerentanan yang lebih proaktif dapat mencegah berbagai eksploitasi sebelum menjadi masalah yang lebih besar [2].

Analisis keamanan CMS juga menggunakan pendekatan berbasis *attack-tree*. "Studi *Cybersecurity of WordPress Platforms. An Analysis Using Attack-Defense Trees Method"* menggunakan metode ini untuk mengidentifikasi cara-cara di mana CMS dapat diserang dan teknik yang dapat digunakan untuk mencegah serangan. Metode pohon serangan memberikan representasi grafis mengenai berbagai jalur serangan yang dapat dieksploitasi oleh peretas, serta tindakan yang dapat diambil untuk mengatasinya. Metode ini memungkinkan penelitian untuk secara lebih sistematis menggambarkan bagaimana ancaman terhadap CMS muncul dan bagaimana tindakan mitigasi dapat diterapkan [14].

B. Metode Penelitian

Penelitian ini mengadopsi pendekatan eksploratif dan evaluatif untuk menganalisis berbagai kerentanan pada sistem manajemen konten (CMS), dengan mengintegrasikan teknik dan alat yang relevan. Untuk memastikan konsistensi, replikasi, dan keandalan hasil, penelitian ini dilaksanakan melalui serangkaian tahapan sistematis, yang meliputi: (1) tinjauan pustaka, (2) persiapan lingkungan uji dan identifikasi versi CMS, (3) simulasi serangan keamanan yang dilakukan secara manual maupun dengan menggunakan alat uji otomatis seperti OWASP ZAP, Burp Suite, Joomscan, dan WPScan, (4) analisis hasil kerentanan yang ditemukan, dan (5) validasi serta pelaporan temuan.



Gambar 1. Diagram Alur Penelitian

Alur kerja penelitian ini, yang digambarkan secara visual pada Gambar 1, terdiri dari beberapa tahapan yang dijelaskan sebagai berikut :

1. Tinjauan pustaka

Tinjauan pustaka merupakan serangkaian kegiatan yang berkaitan dengan metode pengumpulan data, membaca dan mencatat, serta mengolah bahan-bahan penelitian. Tinjauan pustaka dilakukan dengan mengkaji berbagai sumber kredibel, seperti buku akademik, jurnal ilmiah, serta artikel penelitian yang relevan.

2. Identifikasi CMS dan Versi

Sebagai tahap awal, sistem manajemen konten WordPress dan Joomla diinstal dalam lingkungan Docker guna menjamin replikasi konfigurasi standar secara konsisten dan aman [15]. Versi CMS yang digunakan akan diidentifikasi melalui analisis HTTP *headers* dan pemeriksaan menggunakan tools seperti Joomscan, WPScan, dan Wappalyzer dengan tujuan untuk meminimalkan kesalahan dalam mengidentifikasian versi yang terpasang [16]. Penggunaan Docker di sini memberikan keuntungan tambahan berupa isolasi lingkungan yang dapat mempermudah pengelolaan konfigurasi dan uji coba yang lebih aman.

3. Simulasi Serangan Keamanan

Untuk mengevaluasi sejauh mana sistem manajemen konten (CMS) dapat menangani ancaman yang ada, eksperimen simulasi serangan akan dilakukan, termasuk serangan injeksi SQL dan Cross-Site Scripting (XSS). Simulasi ini bertujuan untuk menguji kekuatan pertahanan CMS dalam menangkal jenis serangan yang paling umum ditemukan pada aplikasi web. Pengujian ini akan mengacu pada standar keamanan OWASP Top Ten dan dipadukan dengan analisis temuan pada CVE, guna mengidentifikasi serta menguji kerentanan pada platform seperti WordPress dan Joomla. Dalam simulasi ini, beberapa kategori kerentanaan yang diuji berdasarkan standar OWASP Top Ten adalah **Injection (A3)** yang mencakup pengujian terhadap potensi injeksi SQL dan XSS, **Security Misconfiguration (A5)** yang dievaluasi dengan berfokus pada akses terhadap file sensitif serta konfigurasi server, dan **Vulnerable and Outdated Components (A6)** yang melibatkan evaluasi terhadap komponen-komponen yang sudah usang atau memiliki kerentanan, seperti plugin dan tema pada CMS [17].

Untuk mendukung simulasi serangan ini, sejumlah alat penguji otomatis digunakan untuk mendeteksi dan mengidentifikasi kerentanannya. Secara umum, masing-masing alat memiliki karakteristik dan fokus analisis yang berbeda, seperti OWASP ZAP digunakan untuk melakukan pemindaian otomatis terhadap celah keamanan umum, dengan fokus pada absennya header keamanan penting dan konfigurasi yang kurang optimal. Sementara itu, BurpSuite dimanfaatkan untuk mendeteksi kerentanan berbasis input pengguna melalui teknik semi-otomatis. JoomScan digunakan untuk mengidentifikasi kelemahan pada komponen Joomla, seperti ekstensi usang dan file sensitif, sedangkan WPScan diimplementasikan untuk mendeteksi kerentanan pada plugin, tema, serta informasi pengguna di CMS WordPress.

Deskripsi lebih rinci mengenai fungsi dan mekanisme masing-masing alat disajikan pada subbagian berikut:

1) OWASP ZAP

OWASP Zed Attack Proxy (ZAP) adalah alat pemindai kerentanan yang paling sering digunakan dalam pengujian dan dikembangkan secara open source oleh organisasi OWASP. Karena kemudahan instalasi dan penggunaanya, ZAP dapat digunakan baik oleh pemula serta penguji tingkat lanjut [18]. ZAP dimaksudkan untuk menjadi aplikasi *desktop* yang tersedia pada berbagai platform seperti Windows, Linux dan macOS.

2) Burpsuite

Burpsuite adalah alat pemindai kerentanan yang dapat digunakan untuk melakukan *scanning* pada situs web yang dikembangkan oleh organisasi PortSwigger, kerentanan yang dapat ditemukan menggunakan Burpsuite ini meliputi XSS, SQLi, CSRF, XEE *Injection*, *Directory traversal* dan *Server-side request forgery*. Burpsuite memberikan nilai tertinggi bagi konsultan keamanan dalam hal bug yang ditemukan, kemudahan penggunaan, fleksibilitas lisensi, dan keluasan fitur [19].

3) Joomscan

OWASP Joomla! Vulnerability Scanner (JoomScan) adalah proyek sumber terbuka yang dikembangkan dengan tujuan mengotomatiskan tugas deteksi kerentanan dan jaminan keandalan dalam penerapan Joomla CMS [20].

```
Algorithm JoomScan
Input: Target_URL
Output: List of detected vulnerabilities

- Start
- Initialize Scanner
- Set Target ← Target_URL
- Send HTTP request to Target
- Retrieve HTTP response headers and analyze:
   a. Check Joomla version
   b. Identify installed components and extensions
   c. Detect outdated versions
- Compare findings with vulnerability database (CVE,
Exploit-DB)
- If vulnerabilities found:
      a. Classify severity level (Informational, Low,
Medium, High)
      b. Store vulnerability details
- Generate scan report
- Display results to user
- End
```

4) WPScan

WPScan merupakan alat pemindaian keamanan yang berfokus pada CMS WordPress yang digunakan oleh para profesional dan pengelola *blog* untuk penguji keamanan pada situs web WordPress. WPScan mampu menemukan berbagai jenis kerentanan, mulai dari plugin yang usang hingga tema yang rentan [21].

```
Algorithm WPScan
Input: Target_URL
Output: List of detected vulnerabilities
- Start
- Initialize Scanner
- Set Target ← Target_URL
- Send HTTP request to Target
- Retrieve HTTP response headers and analyze:
  a. Detect WordPress version
  b. Identify installed plugins and themes
  c. Detect outdated versions
- Compare findings with vulnerability databases (CVE,
Exploit-DB)
- If vulnerabilities found:
  a. Classify severity level (Informational, Low, Medium,
   High)
```

```
  b. Store vulnerability details
- Enumerate users (if enabled)
- Generate scan report
- Display results to user
- End
```

4. Analisis data

Data yang diperoleh dari pemindaian, eksperimen, eksplorasi akan dianalisis untuk mengidentifikasi pola kerentanan, tingkat keparahan ancaman, serta tren. Dengan pendekatan ini, diharapkan dapat ditemukan pola yang mendalam mengenai kerentanannya sehingga langkah-langkah mitigasi yang tepat dapat diterapkan untuk mengurangi risiko serangan yang mungkin terjadi pada CMS.

C. Hasil dan Pembahasan

1. Identifikasi versi CMS

WordPress dan Joomla diinstal pada lingkungan Docker guna memastikan konfigurasi standar dapat diuji dan direplikasi secara konsisten. Proses identifikasi versi CMS dilakukan dengan bantuan alat seperti Joomscan, WPScan, dan Wappalyzer untuk meningkatkan akurasi dalam mengenali versi sistem yang diimplementasikan. Proses identifikasi ini mengungkapkan versi dari WordPress yaitu 6.7.1 dan Joomla 5.2.2.

2. Uji kerentanan

Uji kerentanan dilakukan untuk mensimulasikan serangan didunia nyata, yang berfokus pada serangan SQL *Injection* dan *Cross-Site Scripting* (XSS), dua ancaman yang paling umum ditemukan di aplikasi web [22], [23].

1) pengujian SQL Injection

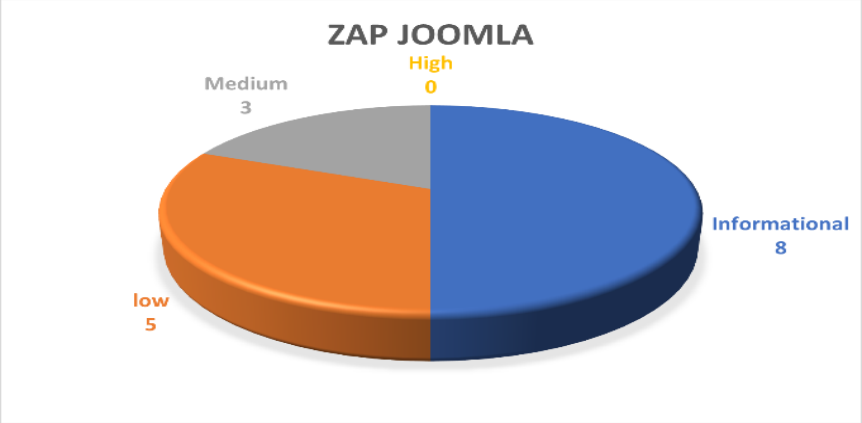
Selama pengujian SQL *Injection*, baik pada WordPress maupun Joomla tidak menunjukan kerentanannya, terutama ketika berinteraksi dengan *form*. Serangan dilakukan dengan otomatis menggunakan SQLmap dan penggunaan *Payloads* terhadap beberapa fitur seperti *registration form*, *login form*, *URL headers*, dan *comment form*.

2) Pengujian XSS

Selama pengujian XSS, pengujian dilakukan pada beberapa fitur seperti *registration form*, *login form*, *URL headers*, dan *comment form*. Serangan dilakukan menggunakan *payloads*, selama pengujian XSS Joomla tidak menunjukan kerentanan sama sekali, sementara WordPress menunjukan kerentanan terhadap XSS ketika dilakukan uji pada fitur *comment form*. XSS yang menjadi kerentanan pada WordPress merupakan XSS yang bertipe XSS *stored*.

3) OWASP ZAP

Gambar 2 memperlihatkan distribusi tingkat risiko kerentanan yang terdeteksi pada situs Joomla berdasarkan hasil pemindaian menggunakan OWASP ZAP.



Gambar 2. Distribusi Risiko Kerentanan Joomla berdasarkan OWASP ZAP

Dari gambar tersebut dapat disimpulkan bahwa sebagian besar kerentanan berada pada tingkat risiko sedang hingga rendah, dengan fokus utama pada absennya header keamanan penting.

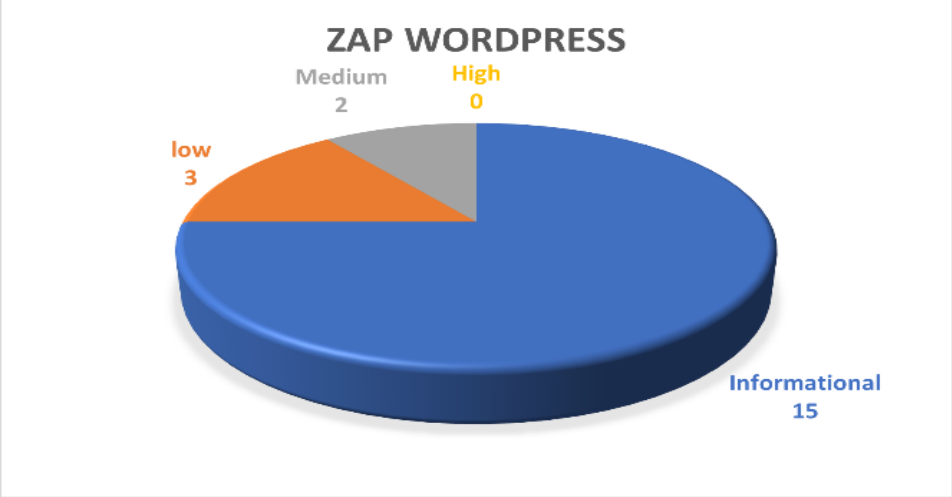
Tabel 1 menyajikan data terperinci mengenai klasifikasi jenis kerentanan yang teridentifikasi pada sistem Joomla, sebagaimana divisualisasikan pada Gambar 2. Klasifikasi indeks tingkat risiko untuk setiap kerentanan didasarkan pada hasil analisis pemindaian menggunakan perangkat lunak OWASP ZAP.

Tabel 1. Jenis Kerentanan Joomla dan Indeks Risiko berdasarkan OWASP ZAP

No	peringatan	indeks tingkat risiko
1	*Absence of Anti-CSRF Tokens*	*Medium*
2	*Content Security Policy (CSP) Header Not Set*	*Medium*
3	*Missing Anti-clickjacking Header*	*Medium*
4	*Cookie No* Http*Only Flag*	*Low*
5	*Cookie without SameSite Attribute*	*Low*
6	*Server Leaks Information*	*Low*
7	*X-Content-Type-Options Header Missing*	*Informational*
8	*Authentication Request*	*Informational*
9	*Charset Mismatch*	*Informational*
10	*Information Disclosure*	*Informational*

Dapat dilihat bahwa kerentanan dominan pada Joomla berkaitan dengan ketidaktersediaan token Anti-CSRF dan header Content Security Policy (CSP), yang berpotensi meningkatkan risiko serangan berbasis web.

Gambar 3 menunjukkan distribusi tingkat risiko kerentanan yang terdeteksi pada situs WordPress hasil pemindaian OWASP ZAP.



Gambar 3. Distribusi Risiko Kerentanan WordPress berdasarkan OWASP ZAP

Berdasarkan hasil tersebut, WordPress cenderung memiliki distribusi kerentanan serupa dengan Joomla, namun ditemukan pula indikasi kerentanan XSS pada atribut HTML yang dapat dikendalikan pengguna.

Tabel 2 menyajikan data terperinci mengenai klasifikasi jenis kerentanan yang teridentifikasi pada sistem WordPress, sebagaimana divisualisasikan pada Gambar 3. Klasifikasi indeks tingkat risiko untuk setiap kerentanan didasarkan pada hasil analisis pemindaian menggunakan perangkat lunak OWASP ZAP.

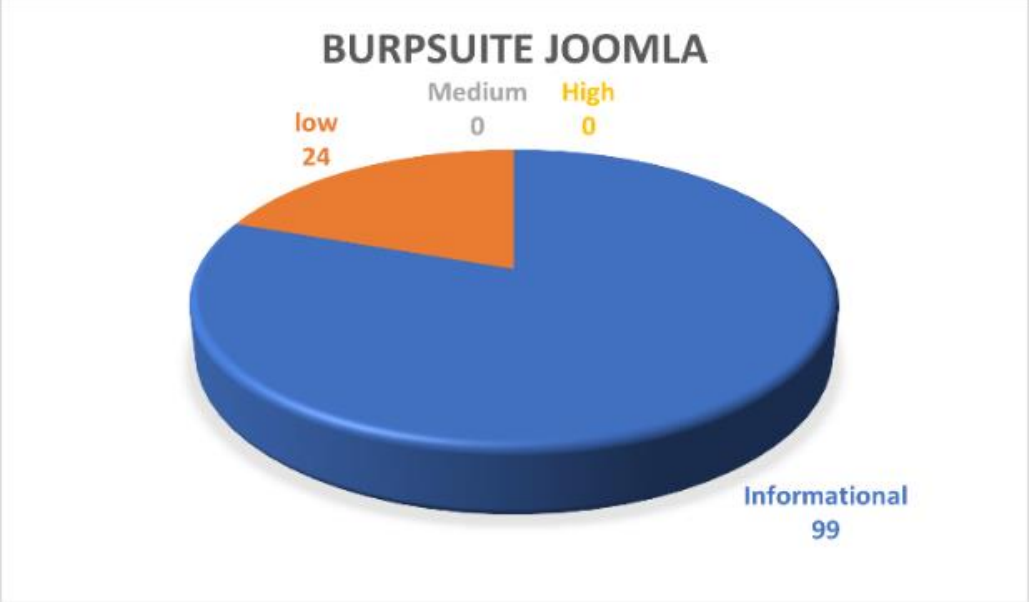
Tabel 2. Jenis Kerentanan WordPress dan Indeks Risiko berdasarkan OWASP ZAP

No	Peringatan	indeks tingkat risiko
1	Content Security Policy (CSP) Header Not Set	Medium
2	Missing Anti-clickjacking Header	Medium
3	Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low
4	Server Leaks Version Information via "Server" HTTP Response Header Field	Low
5	X-Content-Type-Options Header Missing	Low
6	Charset Mismatch	Informational
7	Information Disclosure	Informational
8	Tech Detected	Informational
9	User Controllable HTML Element Attribute (Potential XSS)	Informational

Hasil pemindaian memperlihatkan adanya kekurangan pada pengaturan header keamanan serta adanya potensi manipulasi atribut HTML, yang membuka peluang bagi serangan berbasis XSS.

4) Burpsuite

Gambar 4 memperlihatkan distribusi tingkat risiko kerentanan yang terdeteksi pada situs Joomla berdasarkan hasil pemindaian menggunakan Burpsuite.



Gambar 4. Distribusi Risiko Joomla berdasarkan Pemindaian Burpsuite

Dari gambar tersebut dapat disimpulkan bahwa sebagian besar kerentanan berada pada tingkat risiko sedang hingga rendah, dengan fokus utama pada absennya header keamanan penting.

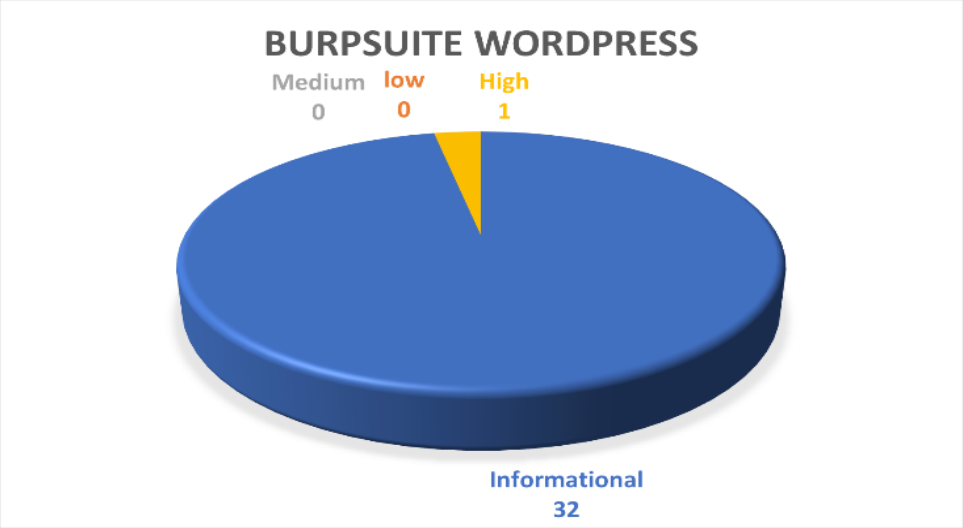
Tabel 3 menyajikan data terperinci mengenai klasifikasi jenis kerentanan yang teridentifikasi pada sistem Joomla, sebagaimana divisualisasikan pada Gambar 4. Klasifikasi indeks tingkat risiko untuk setiap kerentanan didasarkan pada hasil analisis pemindaian menggunakan perangkat lunak Burpsuite.

Tabel 3. Kerentanan Joomla Berdasarkan Pemindaian Burpsuite

No	Kerentanan	indeks tingkat risiko
1	*Client-side* HTTP *parameter pollution (reflected)*	*Low*
2	*Suspicious input transformation (reflected)*	*Informational*
3	*Information Disclosure*	*Informational*

Berdasarkan Tabel 3, sebagian besar kerentanan yang ditemukan bersifat informasional atau berdampak rendah, seperti transformasi input yang mencurigakan dan disclosure informasi. Hal ini mengindikasikan bahwa Joomla, pada konfigurasi standar, relatif aman dari kerentanan kritis berdasarkan pemindaian Burpsuite

Gambar 5 memperlihatkan distribusi tingkat risiko kerentanan yang terdeteksi pada situs WordPress berdasarkan hasil pemindaian menggunakan Burpsuite.



Gambar 5. Distribusi Risiko WordPress Berdasarkan Pemindaian Burpsuite

Berdasarkan hasil tersebut, WordPress cenderung memiliki distribusi kerentanan serupa dengan Joomla, namun ditemukan pula indikasi kerentanan XSS pada atribut HTML yang dapat dikendalikan pengguna.

Tabel 4 berikut mendokumentasikan berbagai jenis kerentanan yang teridentifikasi pada CMS WordPress melalui pemindaian Burpsuite, termasuk tingkat keparahan risiko masing-masing temuan.

Tabel 4. Kerentanan WordPress Berdasarkan Pemindaian Burpsuite

No	peringatan	indeks tingkat risiko
1	*Cross-origin resource sharing*	*High*
2	*Suspicious input transformation (reflected)*	*Informational*
3	*Link manipulation*	*Informational*
4	*Information Disclosure*	*Informational*

Hasil pemindaian memperlihatkan adanya kerentanan berisiko tinggi pada WordPress, khususnya terkait Cross-Origin Resource Sharing (CORS). Di samping itu, beberapa kerentanan informasional seperti suspicious input transformation dan link manipulation turut terdeteksi, memperlihatkan adanya peluang potensi eksploitasi jika tidak dilakukan mitigasi.

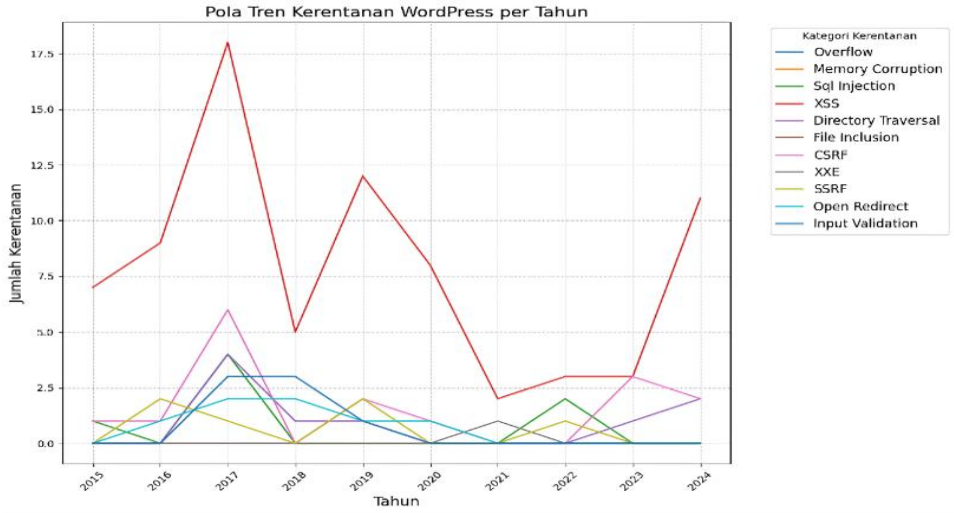
5) Joomscan dan WPScan

Pemindaian menggunakan JoomScan dan WPScan menunjukkan tidak adanya kerentanan kritis pada Joomla dan WordPress, namun ditemukan beberapa informasi terbuka yang perlu diperhatikan untuk mencegah potensi eksploitasi di masa depan.

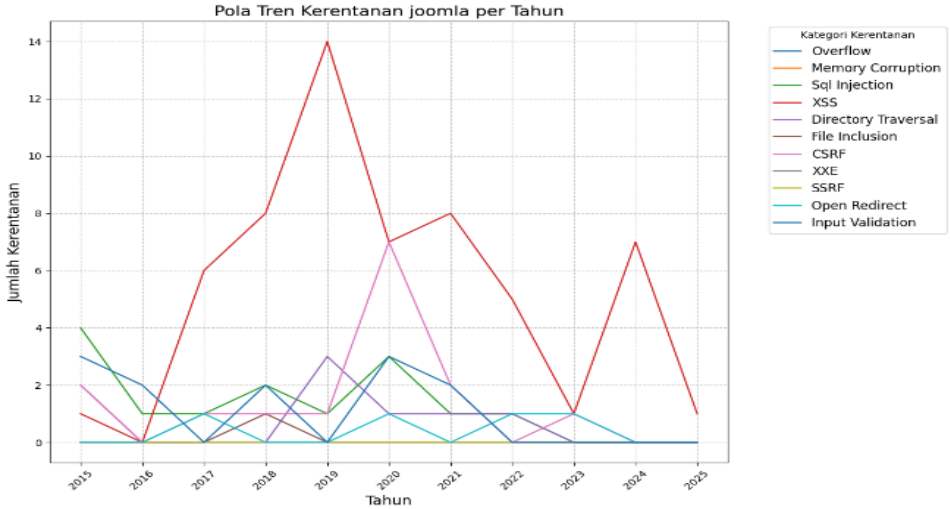
3. Analisis Pola Kerentanan

Berdasarkan data yang diperoleh, pola kerentanan utama pada CMS WordPress dan Joomla didominasi oleh Cross-Site Scripting (XSS) dan Cross-Site Request Forgery (CSRF) sebagai ancaman paling sering dijumpai [22], [23]. XSS mencakup 55.97 persen dari total kerentanan, diikuti oleh *Cross-Site Request Forgery* 12.76 persen. data ini terhitung dari tahun 2015-2025 [22], [23]. Hasil ini sejalan dengan penelitian yang dilakukan oleh Hannes Ekstam Ljusegren (2023) dalam *Vulnerabilities in Outdated Content Management Systems: An Analysis of the Largest WordPress* dan Patryk Zamościński (2020) dalam *Analysis of Security CMS Platforms by Vulnerability Scanners*, yang juga menemukan dominasi XSS dan CSRF sebagai kerentanan utama dalam CMS [9], [24].

Dibandingkan dengan temuan sebelumnya, data dari basis data CVEdetails menunjukkan tren yang konsisten di mana XSS tetap menjadi kerentanan dengan persentase tertinggi setiap tahunnya, sementara CSRF menempati posisi kedua. Pola ini tetap stabil dari tahun ke tahun tanpa adanya perubahan signifikan Gambar 5 dan Gambar 6 menggambarkan tren evolusi jumlah kerentanan yang dilaporkan pada WordPress dan Joomla dalam rentang waktu 2014 hingga 2024.



Gambar 6. Tren Kerentanan WordPress 2014-2024



Gambar 7. Tren Kerentanan Joomla 2014-2025

4. Pemanfaatan Exploit-DB

Penggunaan Exploit-DB melalui alat Searchsploit memungkinkan identifikasi berbagai jenis kerentanan yang terdapat pada versi CMS yang sedang diuji. Alat ini menyaring dan menyajikan eksploitasi yang terdaftar, memberikan gambaran lengkap mengenai potensi celah keamanan yang dapat membahayakan integritas sistem. Hasil yang ditemukan meliputi kerentanan seperti injeksi SQL, XSS, dan CSRF, yang diketahui dapat dieksploitasi oleh pihak yang tidak bertanggung jawab jika tidak ditangani dengan benar.

Tabel 5 menyajikan hasil pemindaian kerentanan pada WordPress 6.7.1 dan tabel 6 menyajikan hasil pemindaian kerentanan pada Joomla 5.2.2 yang terdapat pada database Exploit-DB.

Tabel 5. Data kerentanan Wordpress 6.7.1

No	***Exploit title***
1	*NEX-Forms WordPress plugin < 7.9.7 – Authenticated* SQLi
2	*WordPress Plugin DZS Videogallery <8.60 – multiple vulnerability*
3	*WordPress Plugin iThemes Security < 7.0.3* - SQL *Injection*
4	*WordPress Plugin Rest Google Maps < 7.11.18* SQL *Injection*
5	*WordPress Theme Newspaper 6.7.1 - Privilege Escalation*

Tabel 6. Data kerentanan joomla 5.2.2

No	***Exploit Title***
1	*Joomla! Component com_enmasse 5.1 < 6.4* – SQL *Injection*

D. Simpulan

Penelitian ini bertujuan untuk menganalisis dan mengevaluasi kerentanan yang terdapat pada CMS WordPress dan Joomla melalui metode penetration testing, pemindaian kerentanan, serta eksplorasi database eksploitasi. Hasil penelitian menunjukkan bahwa CMS Joomla, dengan konfigurasi standar, tidak mengindikasikan adanya kerentanan terhadap serangan umum seperti SQL Injection (SQLi), Cross-Site Scripting (XSS), maupun Cross-Site Request Forgery (CSRF). Sebaliknya, pada WordPress dengan konfigurasi standar, ditemukan kerentanan berupa Stored Cross-Site Scripting (Stored XSS) pada fitur komentar. Penggunaan Searchsploit berhasil mengidentifikasi kerentanan yang terdaftar pada masing-masing CMS , konfigurasi standar CMS yang tidak diperbarui dapat menjadi faktor utama dalam meningkatnya tingkat kerentanan. Dari berbagai jenis peringatan yang ditemukan pada aplikasi oleh OWASP ZAP dan Burpsuite, kerentanan seperti CORS, *link manipulation*, *client-side* HTTP *parameter pollution*, dan XSS menunjukan pentingnya sanitasi *input* dan pengaturan yang ketat. Selain itu, penting untuk untuk secara rutin memeriksa dan mengupdate CMS untuk mengurangi potensi eksploitasi.

Untuk penelitian selanjutnya, terdapat beberapa hal yang dapat dilakukan untuk memperluas dan memperdalam pemahaman mengenai kerentanan pada keamanan CMS. Kerentanan secara umum, dampak dan pengaruh plugin terhadap tingkat keamanan CMS masih dapat dieksplorasi. Sejalan dengan temuan tersebut, *plugin* pihak ketiga sering kali menjadi faktor utama yang berkontribusi terhadap munculnya kerentanan seperti diidentifikasi melalui

Searchsploit, Namun, dengan pemilihan dan konfigurasi yang tepat, beberapa plugin juga dapat berkontribusi dalam meningkatkan sistem keamanan.

E. Referensi

[1] E. Ramalingam, "Research Paper on Content Management Systems (CMS): Problems in the Traditional Model and Advantages of CMS in Managing Corporate Websites," 2016. [Online]. Available: http://digitalcommons.harrisburgu.edu/pmgt_dandt/7

[2] S. Bose and A. K. Narayanan, "Security Analysis of CMS based Websites through CMSPY," *INTERNATIONAL CENTER FOR RESEARCH AND RESOURCES DEVELOPMENT*, vol. 4, no. 4, Dec. 2023, doi: 10.53272/icrrd.

[3] M. Md, D. Arzoo, and M. A. Tiwari, "Research study on content management systems (CMS): issues with the conventional model and CMS's benefits for running business websites," 2023. [Online]. Available: www.irjet.net

[4] S. Maruli Panjaitan, R. Naibaho, and E. Penulis Korespondensi, "Jurnal Informatika Dan Rekayasa Komputer (JAKAKOM) Perancangan Forum Diskusi Mahasiswa Berbasis Website (Studi Kasus Universitas Dinamika Bangsa Jambi)," 2022. [Online]. Available: https://ejournal.unama.ac.id/index.php/jakakom

[5] A. Kuzior, I. Tiutiunyk, A. Zielińska, and R. Kelemen, "Cybersecurity and cybercrime: Current trends and threats," *Journal of International Studies*, vol. 17, no. 2, pp. 220–239, 2024, doi: 10.14254/2071-8330.2024/17-2/12.

[6] B. Shteiman, "Why CMS platforms are breeding security vulnerabilities," *Network Security*, vol. 2014, no. 1, pp. 7–9, Jan. 2014, doi: 10.1016/S1353-4858(14)70006-6.

[7] W3Techs, "Usage statistics and market shares of content management systems," W3 techs. Accessed: Jan. 30, 2025. [Online]. Available: https://w3techs.com/technologies/overview/content_management

[8] A. Gustiyono, E. I. Alwi, and S. M. Abdullah, "Analisa Kerentanan Website Terhadap Serangan Cross-Site Scripting (XSS) Metode Penetration Testing," 2024.

[9] H. Ekstam Ljusegren, "Vulnerabilities in Outdated Content Management Systems : An Analysis of the Largest WordPress Websites.," Linköping University, Department of Computer and Information Science, 2023.

[10] P. Jerman Blažič, "Web vulnerability in 2021: large scale inspection, findings, analysis and remedies," 2021.

[11] Mahesh Bhandari, "COMPARISON OF WORDPRESS, JOOMLA AND DRUPAL," TURKU UNIVERSITY OF APPLIED SCIENCES, 2020.

[12] M. Niemietz, M. Korth, C. Mainka, and J. Somorovsky, "Over 100 Bugs in a Row: Security Analysis of the Top-Rated Joomla Extensions," Feb. 2021, [Online]. Available: http://arxiv.org/abs/2102.03131

[13] M. Asaduzzaman, P. P. Rawshan, N. N. Liya, M. N. Islam, and N. K. Dutta, "A vulnerability detection framework for CMS using port scanning technique," in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, Springer, 2020, pp. 128–139. doi: 10.1007/978-3-030-52856-0_10.

[14] G. Petrică, "Cybersecurity of WordPress Platforms. An Analysis Using Attack-Defense Trees Method," in *Cybersecurity of WordPress Platforms. An Analysis Using Attack-Defense Trees Method*, Bucharest: Proceedings of the International Conference on Cybersecurity and Cybercrime, 2022.

[15] R. A. Megantara, F. Alzami, R. A. Pramunendar, and D. P. Prabowo, "PENGEMBANGAN DAN IMPLEMENTASI DOCKER UNTUK MEMAKSIMALKAN UTILITAS SERVER UNIVERSITAS PADA MASA COVID-19," *Transmisi*, vol. 24, no. 2, pp. 48–54, May 2022, doi: 10.14710/transmisi.24.2.48-54.

[16] D. Gupta and M. Vikas Sahni, "A Critical Review of WordPress Security Scanning Tools and the Development of a Next-Generation Solution MSc Research Project MSc in Cybersecurity," National College of Ireland, 2023. [Online]. Available: https://wpbeginner.com/showcase/24-must-have-wordpress-plugins-for-business-websites/

[17] O. Ben Fredj, O. Cheikhrouhou, M. Krichen, H. Hamam, and A. Derhab, "An OWASP Top Ten Driven Survey on Web Application Protection Methods," 2021, pp. 235–252. doi: 10.1007/978-3-030-68887-5_14.

[18] F. P. E. Putra, U. Ubaidi, A. Hamzah, W. A. Pramadi, and A. Nuraini, "Systematic Literature Review: Security Gap Detection On Websites Using Owasp Zap," *Brilliance: Research of Artificial Intelligence*, vol. 4, no. 1, pp. 348–355, Jul. 2024, doi: 10.47709/brilliance.v4i1.4227.

[19] D. R. Mathew and J. Benjamin, "Penetration Testing and Vulnerability Scanning of Web Application Using Burp Suite," *National Conference on Emerging Computer Applications (NCECA)-*, 2021, doi: 10.5281/zenodo.5094090.

[20] OWASP, "Joomscan." Accessed: Jan. 18, 2025. [Online]. Available: https://github.com/OWASP/joomscan

[21] G. T. A. Ramadhani, M. R. R. Steyer, M. H. Maulidan, and A. Setiawan, "Analisis Kerentanan WordPress dengan WPScan dan Teknik Mitigasi," *Journal of Internet and Software Engineering*, vol. 1, no. 4, p. 15, Jun. 2024, doi: 10.47134/pjise.v1i4.2613.

[22] SecurityScorecard, "CVEdetailsJoomla." Accessed: Jan. 07, 2025. [Online]. Available: https://www.cvedetails.com/vendor/3496/Joomla.html

[23] SecurityScorecard, "CVEdetails Wordpress." Accessed: Jan. 07, 2025. [Online]. Available: https://www.cvedetails.com/product/4096/Wordpress-Wordpress.html?vendor_id=2337

[24] P. Zamościński and G. Kozieł, "Analysis of security CMS platforms by vulnerability scanners Badanie bezpieczeństwa wybranych platform CMS za pomocą skanerów podatności," 2020.