Jurnal Riset Sistem dan Teknologi Informasi (RESTIA)
Vol. 3 No. 1 | February 2024
Hal 36-50

Analisis dan Pengujian Kerentanan Website Menggunakan OWASP ZAP

Muhammad Amirul Mu'min[1], Novi Tristanti[2], Galih Pramuja Inngam Fanani[3*]
[1]*Ilmu Komputer, Universitas Muhammadiyah Bima, Bima, Indonesia*
[2]*Informatika, Universitas Muhammadiyah Karanganyar, Karanganyar, Indonesia*
[3]*Sistem dan Teknologi Informasi, Universitas 'Aisyiyah Surakarta, Surakarta, Indonesia.*
[*]*E-mail: galihfanani58@gmail.com*
muhamirul98@gmail.com[1]; novitristanti@umuka.com[2];galihfanani58@gmail.com[3]

Article History
Submitted : Dec 03, 2024
Revised : Dec 14, 2024
Accepted : Dec 23, 2024
Available Online : Jan 02, 2025
Published Regularly : Jan 02, 2025

Kata Kunci: *Internet; Information Systems; OWASP ZAP; Siber; Website*

Keywords: *Internet; Information Systems ; OWASP ZAP; Siber; Website*

Contact
Author
galihfanani58@gmail.com

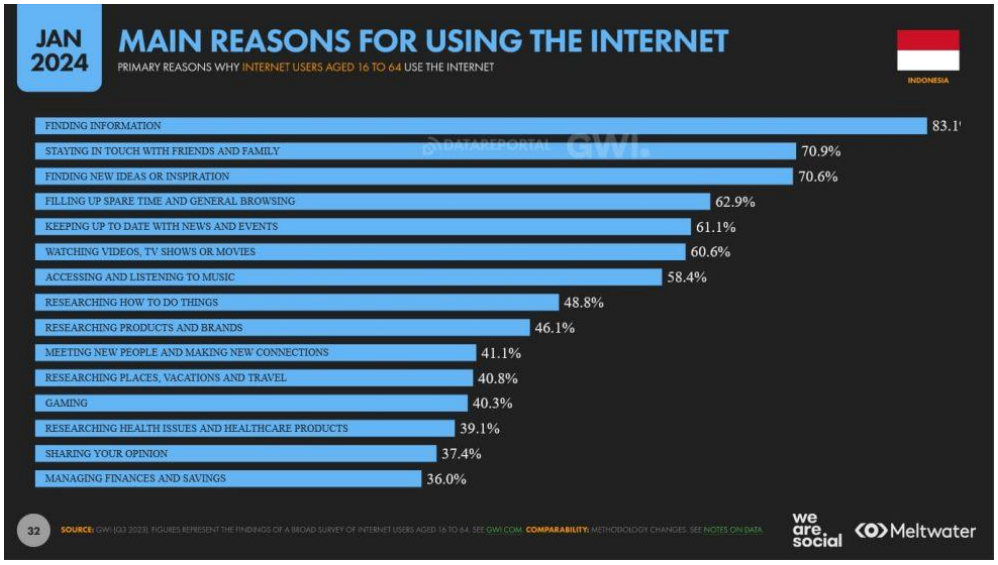
ABSTRAK
Penggunaan internet sedang meningkat, dengan situs web seperti mesin pencari, e-commerce, media sosial, dan portal berita yang sering diakses. Namun, situs web ini sering memiliki celah keamanan yang dapat dieksploitasi untuk ancaman siber. Oleh karena itu penelitian ini bertujuan untuk meningkatkan ketahanan web terhadap serangan siber dan memastikan pengalaman pengguna yang lebih aman, lebih andal dan data pengguna terlindungi. OWASP ZAP adalah alat keamanan yang banyak digunakan yang membantu organisasi mengidentifikasi dan mengatasi kerentanan dalam aplikasi web. Alat ini menawarkan fitur seperti pemindaian otomatis, kemampuan pengujian manual, dan fungsionalitas pelaporan yang komprehensif. Analisis kerentanan berbasis OWASP ZAP membantu mengidentifikasi tingkat keamanan aplikasi web melalui metode pemindaian pasif dan aktif, mendeteksi celah keamanan seperti injeksi SQL, skrip lintas situs, dan konfigurasi yang tidak aman. Temuan kerentanan seperti A01, A03, A04, A05, A06, A08, dan A09 yang mencakup ancaman seperti Cross-Site Scripting (XSS), Clickjacking, dan Man-in-the-Middle menyoroti pentingnya penerapan langkah-langkah mitigasi untuk melindungi keamanan situs web. Penerapan solusi seperti konfigurasi header keamanan (CSP, HSTS, dan X-Frame Options) serta perlindungan terhadap data sensitif sangat penting untuk mencegah eksploitasi. Sehingga dalam pencegahannya diperlukan penerapan protokol enkripsi, pembaruan perangkat lunak secara berkala, pelaksanaan penilaian kerentanan, dan pelatihan karyawan tentang praktik terbaik keamanan siber.

ABSTRACT
Internet usage is on the rise, with websites such as search engines, e-commerce, social media, and news portals being frequently accessed. However, these websites often have security loopholes that can be exploited for cyber threats. Hence this study aims to improve the resilience of the web to cyber attacks and ensure a safer, more reliable user experience and user data is protected. OWASP ZAP is a widely used security tool that helps organizations identify and address vulnerabilities in web applications. The tool offers features

such as automated scanning, manual testing capabilities, and comprehensive reporting functionality. OWASP ZAP-based vulnerability analysis helps identify the security level of web applications through passive and active scanning methods, detecting security loopholes such as SQL injection, cross-site scripting, and insecure configurations. Finding vulnerabilities such as A01, A03, A04, A05, A06, A08, and A09 covering threats such as Cross-Site Scripting (XSS), Clickjacking, and Man-in-the-Middle highlights the importance of implementing mitigation measures to protect website security. Implementing solutions such as security header configuration (CSP, HSTS, and X-Frame Options) and protecting sensitive data are essential to prevent exploitation. Therefore, prevention requires the implementation of encryption protocols, regular software updates, conducting vulnerability assessments, and training employees on cybersecurity best practices.

1. Pendahuluan

Penggunaan internet yang terus meningkat dari tahun ke tahun. Beberapa situs web yang sering diakses oleh pengguna antara lain *search engine, e-commerce*, media sosial, portal berita, dan lainnya [1], [2]. Namun, di balik kemudahan layanan yang diberikan oleh setiap situs web tersebut, ternyata terdapat beberapa masalah celah keamanan [3]. Dengan memanfaatkan celah keamanan tersebut, seseorang dapat menyerang situs web tersebut [4]. Optimalisasi keamanan web merupakan aspek penting untuk memastikan keamanan dan integritas *platform* daring. Menerapkan berbagai langkah keamanan, organisasi dapat melindungi data, pengguna, dan reputasi mereka secara keseluruhan dari ancaman dan serangan siber [5]. Eksplorasi berbagai strategi dan praktik terbaik untuk mengoptimalkan keamanan *web* guna mengurangi risiko dan meningkatkan postur keamanan siber secara keseluruhan [6]. Beberapa strategi utama untuk optimalisasi keamanan web meliputi penerapan protokol enkripsi, memperbarui perangkat lunak dan sistem secara berkala, melakukan penilaian kerentanan, dan melatih karyawan tentang praktik terbaik keamanan siber [7]. Gambar 1. Merupakan alasan-alasan utama pengguna internet di Indonesia



Gambar 1. Alasan-alasan utama pengguna internet di Indonesia

Salah satu strategi utama untuk optimalisasi keamanan web adalah memperbarui perangkat lunak secara berkala dan menambal kerentanan untuk mencegah potensi pelanggaran [8]. Selain itu, menerapkan kontrol akses dan protokol enkripsi yang kuat dapat lebih

melindungi informasi sensitif dari akses yang tidak sah [5]. Proses mendapat informasi tentang ancaman yang muncul dan terus memantau aktivitas yang mencurigakan, organisasi dapat secara efektif melindungi aset web mereka dan menjaga lingkungan daring yang aman [9]. Pada akhirnya, memprioritaskan optimalisasi keamanan web sangat penting untuk mengurangi risiko dan memastikan keberlangsungan operasi digital [10]. Salah satu aspek penting dari pengoptimalan keamanan web adalah penerapan OWASP ZAP (*Open Web Application Security Project Zed Attack Proxy*) [11]. OWASP ZAP adalah alat keamanan yang banyak digunakan yang membantu organisasi mengidentifikasi dan mengatasi kerentanan dalam aplikasi web. Dengan memanfaatkan OWASP ZAP, dalam bidang bisnis dapat menjaga keamanan menyeluruh, mengidentifikasi potensi kelemahan, dan mengambil tindakan proaktif untuk meningkatkan postur keamanan keseluruhan aset web mereka [12]–[14]. Alat ini menawarkan berbagai fitur, termasuk pemindaian otomatis, kemampuan pengujian manual, dan fungsionalitas pelaporan yang komprehensif [4].

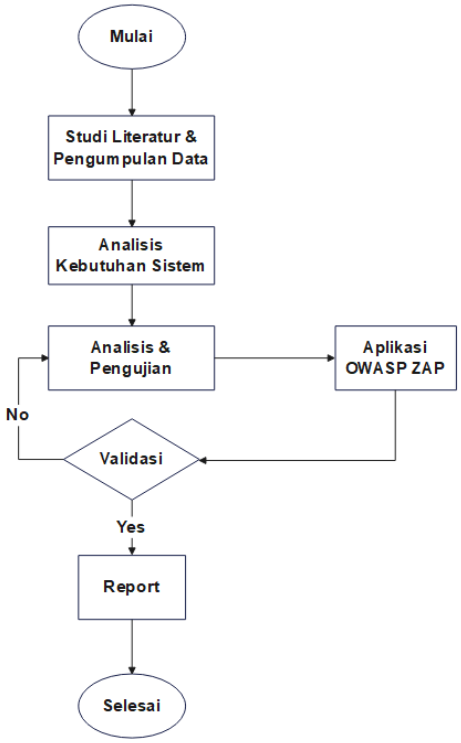
Penelitian serupa dilakukan dengan judul *Penetration Testing on Mail Server Website Using The OWASP Method*. penelitian ini bertujuan melakukan uji penetrasi pada domain mail *server*, mail.umtk.sch.id, dengan menggunakan tools OWASP Zap dan Acunetix [15]. Penelitian kedua berjudul *Open Worldwide Application Security Project (OWASP) Operating Systems*. Penelitian ini bertujuan pengembang dapat membuat aplikasi daring yang lebih aman, sementara pakar keamanan dapat menggunakannya untuk menemukan dan memperbaiki masalah dengan lebih efisien [16]. Penelitian ketiga tentang *Web Application Security Education Platform Based on OWASP API Security Project*. Penelitian ini bertujuan memberikan 10 tantangan kerentanan API ke *platform* dengan 3 tingkat risiko keparahan yang berbeda yang dapat dimanfaatkan menggunakan alat seperti Burp Suite, SQLMap, dan JWTCat [17]. Serta penelitian keempat *Website Vulnerability Testing and Analysis of Internet Management Information System Using OWASP*, dengan tujuan melakukan teknik *gray box penetration testing* dengan menggunakan metode OWASP dan *tool* OWASP ZAP [11].

Bagaimanapun keamanan sistem informasi yang lemah dapat mengganggu infrastruktur organisasi dan bisnis. Berbagai serangan seperti *Malware*, Eksploitasi, dan Injeksi database sering terjadi di internet [7]. Upaya mengurangi risiko ini, dilakukan *penetration testing* (Pentest) guna menguji keamanan web secara legal dengan metode yang menyerupai hacker [18]. Analisis kerentanan berbasis OWASP ZAP menggunakan *tools security* membantu mengidentifikasi tingkat keamanan aplikasi web. Melalui metode pemindaian pasif dan aktif serta mampu mendeteksi celah keamanan seperti SQL *Injection*, *Cross-Site Scripting* (XSS), dan konfigurasi yang tidak aman [16]. Selain itu, proses pengujian dilakukan secara sistematis dengan teknik *crawling*, *fuzzing*, serta integrasi dalam *pipeline* CI/CD untuk memastikan keamanan aplikasi sejak tahap pengembangan. Implementasi OWASP ZAP pada penelitian ini bertujuan untuk meningkatkan ketahanan web terhadap serangan siber dan memastikan pengalaman pengguna yang lebih aman serta andal.

2. Metode Penelitian

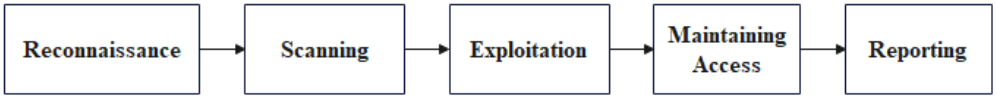
2.1. Kerangka Kerja

Kerangka kerja yang diterapkan dalam menganalisis kerentanan suatu *website* menggunakan aplikasi OWASP ZAP. Penelitian ini menggunakan website granicus.com/goverment-website-design/ sebagai obyek uji, untuk mengevaluasi tingkat kerentanannya terhadap berbagai jenis serangan menggunakan OWASP ZAP. Seperti yang ditunjukkan pada Gambar 2.



Gambar 2. Alur Proses Peneletian menggungakan OWASP ZAP

Tahap pertama dalam penelitian ini adalah studi literatur dari berbagai sumber, seperti jurnal dan buku, untuk memahami lebih dalam tentang metode serangan dan potensi kerentanan pada situs *website* tersebut [19]. Pengumpulan data untuk memperoleh informasi yang relevan guna mencapai tujuan penelitian. Analisis kebutuhan sistem bagian menyiapkan segala perlengkapan yang mendukung penelitian seperti alat dan bahan. Analisis dan pengujian bagian melakukan testing pada sistem *website*. Setelah itu, dilakukan analisis kerentanan menggunakan OWASP ZAP untuk mengidentifikasi celah keamanan dan mencari solusinya [8], [20]. Tahapan proses dalam penelitian ini mengacu pada kerangka kerja OWASP ZAP yang berfokus pada keamanan website. Penggunaan tersebut dapat membantu dalam mengidentifikasi kerentanan sistem secara detail. Berikut ini adalah tahapan kerangka kerja OWASP ZAP sebagaimana diilustrasikan pada Gambar 3.



Gambar 3. Tahapan *Open Web Application Security Project*

Penjelasan setiap tahapan dalam kerangka kerja OWASP seperti pada Gambar 3. terdiri dari lima bagian utama yang saling berhubungan, yaitu:

1. *Reconnaissance*, yaitu tahap awal pengumpulan data dan informasi terkait sistem web yang akan ditembus.
2. *Scanning*, yaitu melakukan pemindaian untuk mendapatkan informasi lebih dalam dan mengidentifikasi kerentanan.
3. *Exploitation*, yaitu melakukan penyerangan terhadap sistem dengan menggunakan data dan informasi yang diperoleh pada saat scanning.
4. *Maintaining Access*, yaitu menjaga akses yang diperoleh dengan cara memasukkan *backdoor* melalui kerentanan pada sistem web.
5. *Reporting*, yaitu menulis laporan yang menjelaskan hasil pengujian disertai dengan rekomendasi dan solusinya.

2.2. Alat dan Bahan

Proses penelitian menggunakan alat yang mencakup perangkat lunak dan perangkat keras yang mendukung proses analisis serta pengujian, sementara bahan merujuk pada data atau sumber daya yang digunakan dalam eksperimen alat dan bahan yang tepat sangat penting untuk memastikan hasil yang akurat dan sesuai dengan tujuan yang ingin dicapai [21]. Tahap ini merupakan persiapan (perencanaan) sebelum pengujian penetrasi pada aplikasi situs web [22]. Data pengguna diperlukan untuk aplikasi situs web manajemen internet Kragan. Pengujian peralatan pendukung menggunakan *computer assisted audit technique* (CAAT) dan alat OWASP ZAP seperti yang ditunjukkan pada Tabel 1.

Tabel 1. Perangkat keras dan lunak

Tool Name	Spesification
Laptop	OS: Windows 10 64 bit Processor : Intel Core i7-8565U quad-core 2,8GHz RAM : 16.0 GB DDR4 VGA : NVidia GeForce MX150 VRAM 2GB GDDR5 SSD : 500GB
OWASP ZAP	Version 2.11.1
Internet Koneksi	Up to 100Mbps
Web Browser	Google Chrome Version 105.0.5

3. Hasil dan Pembahasan

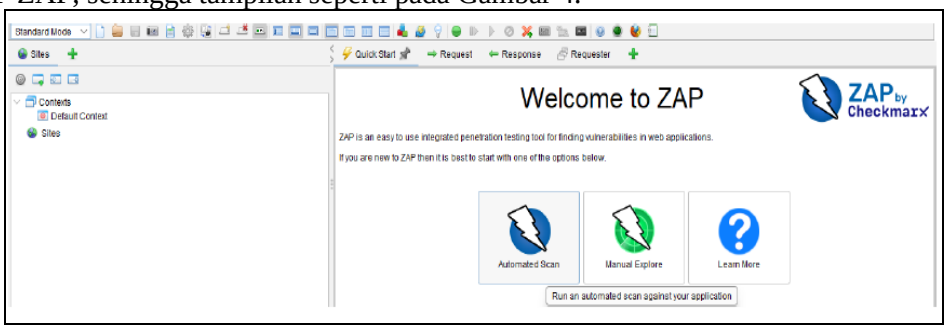
Penelitian ini menerapkan OWASP ZAP untuk meningkatkan optimalisasi pada web dengan mengidentifikasi dan mengatasi berbagai kerentanan keamanan yang dapat dieksploitasi oleh pihak tidak bertanggung jawab.

3.1. *Reconnaissance*

Peneliti melakukan pengumpulan data dan informasi terkait sistem web yang akan dilakukan pengujian langsung pada website www.granicus.com/goverment-website-design/ karena sudah mengetahui mana yang ingin diuji keamanannya.

3.2. *Scanning*

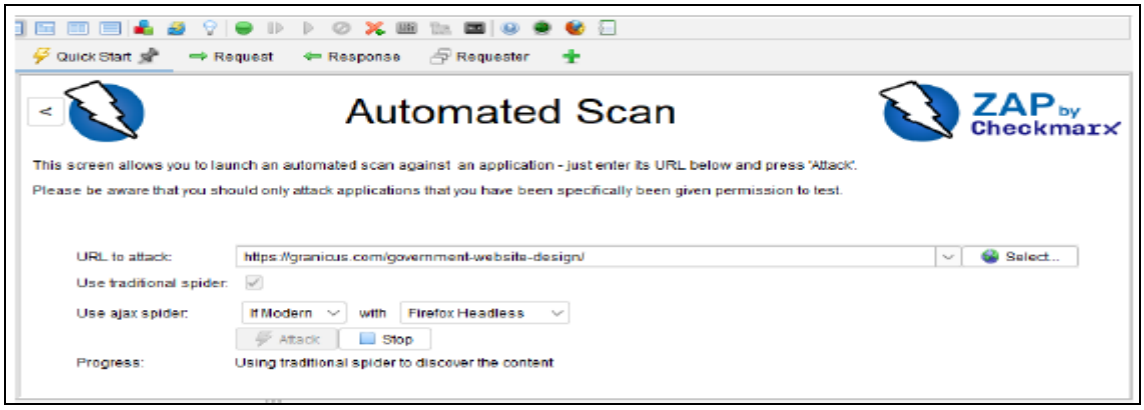
Tahapan ini melakukan pemindaian untuk mendapatkan informasi lebih dalam dan mengidentifikasi kerentanan. Proses ini melibatkan program OWASP ZAP ketika tampilan awal muncul. Setelah itu, pengguna harus mengklik *Automatic Scan* pada kolom *Welcome* to OWASP ZAP, sehingga tampilan seperti pada Gambar 4.



Gambar 4. Tampilan OWASP ZAP

1. *Input URL Website*

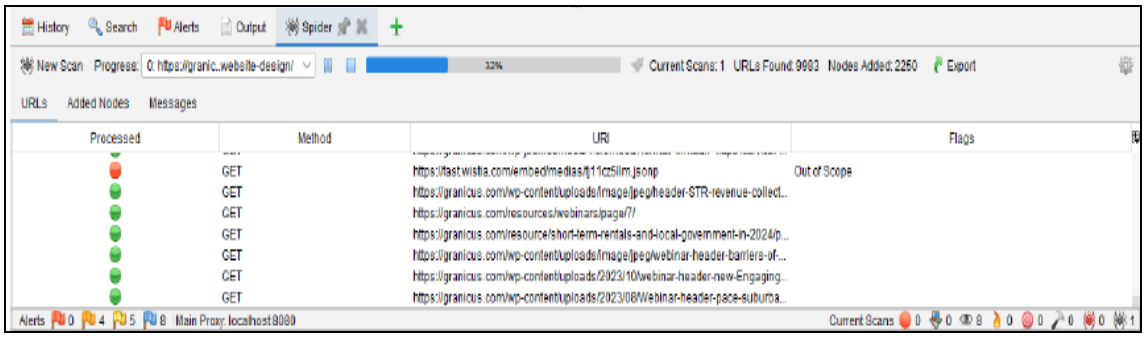
Pada proses ini, memasukkan tautan *website* yang akan diuji ke dalam kolom URL untuk serangan, memilih *Use traditional spider* dan *Use* AJAX *spider for scanning assistance*. Setelah itu, mengklik tombol serangan untuk memulai pemindaian otomatis. ZAP kemudian memeriksa situs *website* yang telah dimasukkan. Seperti pada Gambar 5. Merupakan tampilan *automated scan* ZAP.



Gambar 5. Tampilan *Automated Scan* pada OWASP ZAP.

2. Scanning Website

Kemudian mengamati proses *scanning*. Beberapa hal yang perlu diperhatikan yaitu bar kemajuan menunjukkan bahwa pemindaian telah mencapai 32%, daftar URL yang diproses mencakup metode HTTP dan URL yang dianalisis, serta ikon berwarna menunjukkan status (hijau untuk berhasil, merah untuk kesalahan). Selain itu, URL yang ditandai sebagai "*Out of Scope*" tidak termasuk dalam pemindaian, dan bagian pesan menampilkan log atau notifikasi tambahan yang berguna untuk *debugging*, seperti yang ditunjukkan pada Gambar 6.



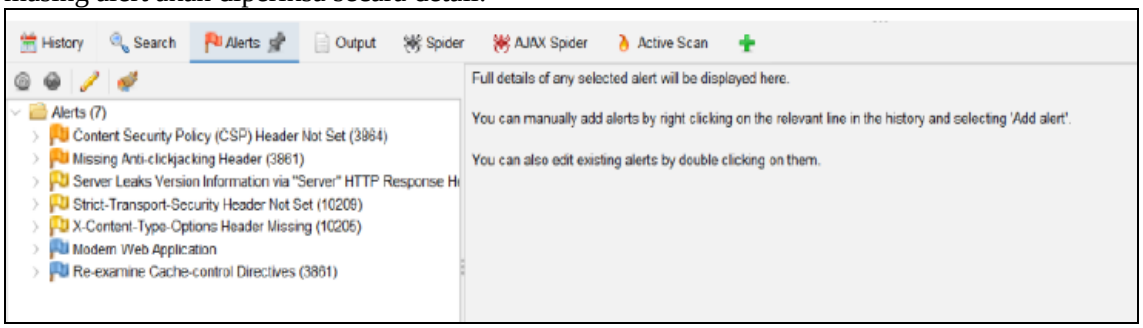
Gambar 6. Proses *Scanning* URL *Website*

Gambar 6. Menunjukkan daftar peringatan yang terdeteksi setelah proses pemindaian selesai. Sehingga dapat melihat informasi terperinci tentang setiap peringatan yang tercantum, termasuk kerentanan yang ditemukan, tingkat keparahan, dan perbaikan yang disarankan. Selain itu, dengan mengeklik dua kali pada peringatan, dapat menambahkan peringatan baru atau mengedit peringatan yang sudah ada. Seperti yang ditunjukkan pada Gambar 5.

3.3. *Exploitation*

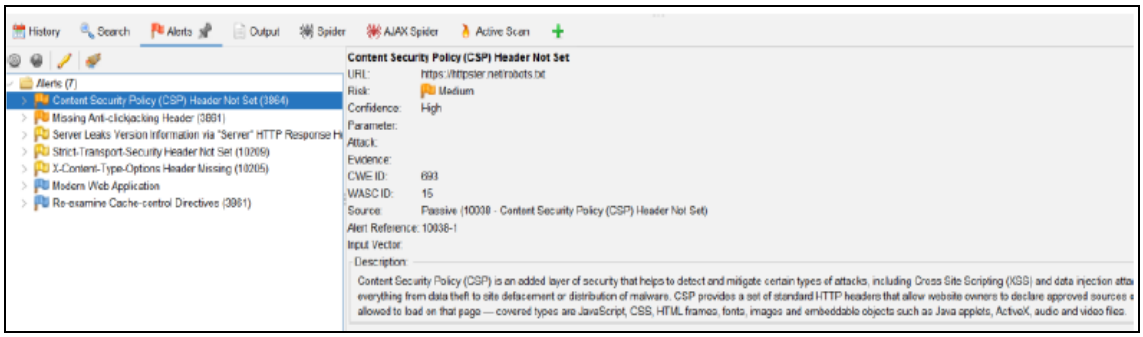
Setelah dilakukan penyerangan atau *testing* terhadapa web tersebut, maka akan diperoleh sebuah alert, yang menunjukana beberapa kerentanan. Gambar 7. Menunjukan hasil ancaman ada 7 alert yakni *content security policy*, *Missing anti-clik*, *Server Leaks*, *Strict Transport*

Security, X-content, Modem Web Application, dan *Re-examine Cache*. Selanjutnya masing-masing alert akan diperiksa secara detail.



Gambar 7. Hasil Ancaman dari Proses Scanning Website

Pada tahap akhir, setelah menganalisis kerentanan pada tab peringatan, sistem akan menampilkan rincian spesifik mengenai masalah keamanan tersebut. Rincian ini mencakup URL yang terdampak, tingkat keyakinan terhadap keberadaan kerentanan, jenis serangan yang berpotensi terjadi, serta penjelasan dan lokasi input yang terpengaruh. Informasi ini memungkinkan untuk mengambil tindakan guna memperbaiki kerentanan dan meningkatkan keamanan situs web yang dianalisis. Seperti yang ditunjukkan pada Gambar 8.



Gambar 8. Merupakan Rician dari Alert

3.4. *Maintaining Access*

Perlu dilakukan upaya untuk menjaga akses agar peretas mendapatkan akses tidak sah ke sistem tanpa terdeteksi, sering kali dengan memanfaatkan kerentanan keamanan dalam aplikasi atau server. Salah satu cara umum backdoor yang disisipkan adalah melalui eksploitasi kelemahan dalam kode, seperti *Remote Code Execution* (RCE) atau celah SQL *Injection* yang memungkinkan penyusupan skrip berbahaya. Misalnya, dalam sebuah serangan terhadap *platform* CMS yang tidak diperbarui, peretas dapat mengeksploitasi kerentanan untuk mengunggah *file* berbahaya yang berfungsi sebagai *backdoor*, memungkinkan mereka mengakses *server* kapan saja tanpa perlu kredensial sah. Selain itu, celah dalam autentikasi yang lemah, seperti penggunaan kata sandi default atau mekanisme otorisasi yang salah konfigurasi, juga dapat dimanfaatkan untuk menanam *backdoor*. Setelah masuk, penyerang dapat mencuri data, memodifikasi sistem, atau bahkan menggunakannya sebagai pijakan untuk serangan lebih lanjut. Oleh karena itu, deteksi dini dan penerapan langkah-langkah keamanan seperti pembaruan rutin, pemantauan log aktivitas, dan penggunaan *firewall* aplikasi web (WAF) sangat penting untuk mencegah ancaman ini.

3.5. *Report*

1. *Summary of Alerts*

Summary of Alerts adalah ringkasan dari ancaman yang terdeteksi melalui pemindaian menggunakan OWASP-ZAP. Ringkasan ini mencakup berbagai jenis kerentanan yang ditemukan, tingkat keparahannya, serta detail terkait potensi risiko keamanan pada sistem atau aplikasi yang diuji. Dengan adanya ringkasan ini, pengguna dapat lebih mudah mengidentifikasi dan mengambil langkah yang diperlukan untuk memperbaiki serta meningkatkan keamanan sistem seperti yang ditunjukkan pada Tabel 2.

Tabel 2. *Summary of Alert*

Risk Level	Number of Alerts
High	0
Medium	2
Low	3
Information	2

Alerts menampilkan daftar nama ancaman beserta tingkat keparahannya. Selain itu, *Alerts* juga mencantumkan jumlah kejadian dari setiap ancaman yang berhasil terdeteksi melalui pemindaian menggunakan OWASP-ZAP, seperti yang ditunjukkan pada Tabel 3.

Tabel 3. Rincian *Alert*

Nama	Risk Level	Number of Instances
Content Security Policy (CSP) Header Not Set	Medium	3964
Missing Anti-clikjacking Header	Medium	3861
Server Leaks Versio Information via ”Server” HTTP Response Header	Low	3280
Strict- Transport-Security Header Not Set	Low	10209
X-Content-Type-Options Header Missing	Low	10205
Modem Web Application	Information	-
Re-examine Cache-control Directives	Information	3861

Tabel 3. Menampilkan daftar nama ancaman yang terdeteksi selama pengujian serta jumlah kejadian dari setiap ancaman yang berisiko diretas atau mengalami pencurian data. Hasil pengujian menunjukkan bahwa jumlah kejadian tertinggi mencapai 10.209, dengan ancaman bernama *Strict-Transport-Security Header Not Set*. Ancaman ini tergolong dalam kategori *Low*, yang berarti memiliki tingkat risiko rendah. Berdasarkan rincian tersebut ditemukan kerentanan berkode A01, A03, A04, A05, A06, A08, dan A09, yang selanjutnya akan dijelaskan melalui tabel 4.

2. Dampak dan Solusi mengatasi kerentanan

Berdasarkan *alert* yang ditemukan, solusi untuk mengatasi kerentanan yang terdeteksi serta langkah mitigasinya dapat merujuk pada rekomendasi dari OWASP ZAP. Solusi ini mencakup tindakan pencegahan, perbaikan konfigurasi, serta penerapan praktik keamanan yang lebih baik untuk mengurangi risiko eksploitasi. Dengan memahami ancaman yang terdeteksi,

pengguna dapat mengambil langkah yang tepat guna meningkatkan keamanan sistem dan melindungi data dari potensi serangan. seperti yang ditunjukkan pada Tabel 4.

Tabel 4. Migitasi Berdasarkan *Alert* Kerentanan

Kerentanan	Alert	Deskripsi	Dampak	Mitigasi
A01 Broken Access Control	Data Leak-Testing Error Message	Kontrol Akses Rusak adalah kerentanan keamanan yang terjadi ketika aplikasi tidak mengelola izin akses pengguna dengan benar.	**Keamanan Data**: Kontrol akses yang rusak dapat menyebabkan kebocoran data sensitif. Informasi pribadi atau rahasia dapat diakses oleh pihak yang tidak berwenang. **Kerugian Finansial**: Kebocoran data dapat mengakibatkan denda dari regulator dan kerugian finansial akibat tindakan hukum atau perbaikan sistem. **Reputasi**: Perusahaan yang mengalami pelanggaran keamanan dapat kehilangan kepercayaan pelanggan dan reputasi di pasar. **Gangguan Operasional:** Akses yang tidak terkontrol dapat mengganggu operasional bisnis, menyebabkan downtime dan mengganggu layanan kepada pelanggan.	**Audit Keamanan Rutin**: Melakukan audit secara berkala untuk mengevaluasi dan memperbaiki kontrol akses. **Pelatihan Karyawan**: Memberikan pelatihan keamanan siber kepada karyawan untuk mengenali potensi ancaman dan praktik terbaik dalam menjaga keamanan informasi. **Penggunaan Teknologi Enkripsi**: Mengimplement asikan enkripsi untuk melindungi data sensitif, bahkan jika akses tidak terkontrol. **Penegakan Kebijakan Akses:** Menetapkan dan menegakkan kebijakan akses yang ketat, termasuk penggunaan autentikasi multi-faktor. **Pemantauan dan Deteksi:** Menggunakan sistem pemantauan untuk mendeteksi aktivitas mencurigakan dan merespons dengan cepat.
A02			Kerentanan A02 yang dimaksud tidak tersedia atau tidak ditemukan pada tautan ini.	**Validasi Input**: Pastikan semua data yang diterima dari pengguna divalidasi dan disaring. **Pengaturan CORS:** Konfigurasi Cross-Origin Resource Sharing (CORS) dengan ketat untuk membatasi akses. **Pembaruan Rutin**: Selalu perbarui perangkat lunak dan pustaka untuk menutup celah keamanan. **Audit Keamanan**: Lakukan audit keamanan secara berkala

A03 Injection	Identifikasi Pengalihan Signifikan (Risiko Kebocoran Data Sensitif	Injeksi adalah jenis kerentanan keamanan yang terjadi ketika data yang tidak diverifikasi atau tidak dibersihkan dimasukkan ke dalam perintah atau kueri yang dijalankan oleh aplikasi	**Kebocoran Data**: Serangan injection, seperti SQL injection, dapat mengakibatk an akses tidak sah ke basis data, mengakibatkan kebocoran informasi sensitif. **Kerugian Finansial**: Perusahaan dapat mengalami kerugian finansial akibat pemulihan sistem, denda, dan kehilangan pendapatan akibat gangguan layanan. **Kerusakan Reputasi**: Serangan yang berhasil dapat merusak reputasi perusahaan, mengurangi kepercayaan pelanggan dan mitra bisnis. **Pengaruh Operasional**: Serangan injection dapat menyebabkan gangguan operasional, mengakibatkan downtime dan menghambat produktivitas	**Validasi Input**: Selalu memvalidasi dan membersihkan input dari pengguna untuk mencegah injeksi kode berbahaya. **Prepared Statements**: Menggunakan prepared statements atau parameterized queries untuk interaksi dengan basis data, yang dapat membantu menghindari SQL injection **Penggunaan Framework Keamanan**: Menggunakan framework yang memiliki fitur keamanan built- in dapat membantu melindungi aplikasi dari serangan injection. **Audit dan Pengujian Keamanan**: Melakukan audit keamanan dan pengujian penetrasi secara berkala untuk mengidentifikasi kerentanan . **Pembaruan Rutin**: Memastikan semua sistem dan perangkat lunak diperbarui untuk mengatasi kerentanan yang diketahui.
A04 Insecure Design	PII (Personal Identifica tion Data) Leak	Desain Tidak Aman adalah kerentanan yang timbul dari keputusan desain yang tidak aman dalam pengembangan perangkat lunak.	**Kerentanan Sistem**: Desain yang tidak aman dapat menciptakan celah yang dapat dieksploitasi oleh penyerang, mengakibatkan kebocoran data atau akses tidak sah. **Kehilangan Data**: Data sensitif dapat hilang atau dicuri akibat kelemahan dalam desain sistem. **Biaya Pemulihan**: Mengatasi serangan yang berhasil dapat memerlukan biaya tinggi untuk pemulihan dan perbaikan sistem. **Kerusakan Reputasi**: Perusahaan yang mengalami pelanggaran akibat desain yang tidak aman dapat kehilangan kepercayaan pelanggan dan mitra bisnis. **Kompleksit as**	**Prinsip Desain Keamanan**: Mengintegrasikan prinsip-prinsip desain keamanan sejak awal pengembangan sistem, termasuk pengujian risiko dan analisis ancaman. **Pengujian Keamanan**: Melakukan pengujian keamanan secara berkala, termasuk pengujian penetrasi dan analisis statis untuk mengidentifikasi potensi kerentanan. **Pelatihan Tim**: Memberikan pelatihan kepada tim pengembang tentang praktik terbaik dalam desain yang aman. **Revisi Kode**: Melakukan code review secara berkala untuk memastikan bahwa desain dan implementasi mengikuti standar keamanan. **Feedback Pengguna**: Mengumpulkan umpan balik

			Pengelolaan: Desain yang tidak aman sering kali menambah kompleksitas dalam pengelolaan dan pemeliharaan sistem, meningkatkan risiko kesalahan manusia	dari pengguna untuk mengidentifikasi area desain yang rentan dan memperbaikinya
A05 Security Misconfiguration	Internal Server Issues	Kesalahan Konfigurasi Keamanan adalah kerentanan yang terjadi ketika pengaturan keamanan aplikasi, server, atau jaringan tidak dikonfigurasi dengan benar	**Akses Tidak Sah**: Konfigurasi yang salah dapat memungkinkan akses tidak sah ke sistem atau data, mengakibatkan kebocoran informasi sensitif. **Kerentanan Sistem**: Kesalahan konfigurasi dapat menciptakan celah keamanan yang dapat dieksploitasi oleh penyerang, memperbesar risiko serangan. **Downtime dan Gangguan Layanan**: Misconfiguration dapat menyebabkan gangguan operasional, downtime, atau bahkan kegagalan sistem. **Biaya Pemulihan**: Memperbaiki kesalahan konfigurasi yang menyebabkan pelanggaran keamanan dapat memerlukan biaya yang signifikan. **Kerusakan Reputasi**: Serangan yang berhasil akibat misconfiguration dapat merusak reputasi perusahaan dan mengurangi kepercayaan pelanggan	**Penerapan Kebijakan Konfigurasi**: Menetapkan dan mendokumentasikan kebijakan konfigurasi keamanan yang jelas untuk semua sistem. **Audit dan Peninjauan Rutin**: Melakukan audit keamanan dan peninjauan rutin konfigurasi untuk memastikan kepatuhan terhadap kebijakan yang ditetapkan. **Penggunaan Alat Otomatisasi**: Menggunakan alat otomatisasi untuk mendeteksi dan memperbaiki kesalahan konfigurasi secara proaktif. **Pendidikan dan Pelatihan**: Memberikan pelatihan kepada tim IT dan pengembang tentang praktik terbaik dalam konfigurasi keamanan. **Penerapan Prinsip Least Privilege**: Mengatur hak akses hanya untuk pengguna yang benar-benar memerlukan untuk menjalankan tugas mereka, mengurangi risiko akses tidak sah.
A06			Kerentanan A06 yang dimaksud tidak tersedia atau tidak ditemukan pada tautan ini	**Enkripsi Data Sensitif**: Gunakan enkripsi untuk melindungi data sensitif saat penyimpanan dan transmisi. **Pengendalian Akses**: Terapkan kontrol akses yang ketat untuk membatasi siapa yang dapat mengakses data. **Pencatatan dan Pemantauan**: Implementasikan sistem pencatatan dan pemantauan

			untuk mendeteksi akses yang tidak sah.
A07		Kerentanan A07 yang dimaksud tidak tersedia atau tidak ditemukan pada tautan ini	**Pemeriksaan Input**: Selalu periksa dan sanitasi input yang diterima dari pengguna untuk mencegah injeksi. **Penggunaan Parameterisasi**: Gunakan pernyataan parameter untuk query database untuk menghindari injeksi SQL. **Pembaruan Perangkat Lunak**: Pastikan semua komponen perangkat lunak diperbarui untuk menutup celah keamanan.
A08 Sofware and Data Integrity Failures	Kegagalan Integritas Perangkat Lunak dan Data adalah kerentanan yang terjadi ketika aplikasi gagal menjaga integritas perangkat lunak dan data yang diprosesnya	**Kehilangan Data**: Kegagalan integritas dapat menyebabkan hilangnya data penting, yang berpotensi mengganggu operasi bisnis. **Kebocoran Informasi**: Data yang tidak terjamin integritasnya dapat diubah atau diakses oleh pihak yang tidak berwenang, mengakibatkan kebocoran informasi sensitif. **Resiko Keputusan Salah**: Data yang tidak akurat dapat menyebabkan pengambilan keputusan yang salah, merugikan strategi dan operasi perusahaan. **Gangguan Operasional**: Kegagalan integritas dapat menyebabkan sistem tidak berfungsi dengan baik, mengakibatkan downtime dan mengganggu layanan kepada pelanggan. **Kerusakan Reputasi**: Kegagalan dalam menjaga integritas data dapat merusak kepercayaan pelanggan dan reputasi perusahaan di pasar	**Implementasi Kontrol Versi**: Menggunakan sistem kontrol versi untuk melacak perubahan pada perangkat lunak dan data, memastikan bahwa perubahan dapat diaudit. **Penerapan Algoritma Hashing**: Menggunakan algoritma hashing untuk memverifikasi integritas data dan mendeteksi perubahan yang tidak sah. **Backup dan Pemulihan**: Melakukan backup data secara rutin dan memiliki rencana pemulihan bencana untuk meminimalkan dampak kehilangan data. **Audit dan Pemantauan Berkala**: Melakukan audit dan pemantauan sistem secara berkala untuk mendeteksi dan memperbaiki masalah integritas dengan cepat. **Pelatihan Karyawan**: Memberikan pelatihan kepada staf tentang pentingnya integritas data dan bagaimana menjaga agar data tetap akurat dan aman.
A09		Kerentanan A09 yang dimaksud tidak tersedia atau tidak ditemukan pada tautan ini	**Autentikasi yang Kuat**: Gunakan metode autentikasi yang kuat, seperti otentikasi dua faktor (2FA). **Manajemen Sesi yang Aman**: Pastikan sesi

		pengguna dikelola dengan aman, termasuk pengaturan timeout dan pengacakan ID sesi. **Validasi Token**: Selalu validasi token autentikasi dan sesi untuk mencegah penyalahgunaan.
A10	Kerentanan A10 yang dimaksud tidak tersedia atau tidak ditemukan pada tautan ini	**Pembaruan Rutin:** Selalu memperbarui perangkat lunak, kerangka kerja, dan pustaka untuk menutup celah keamanan. **Konfigurasi yang Aman:** Pastikan konfigurasi server dan aplikasi aman dan sesuai dengan praktik terbaik. **Penggunaan Firewall:** Terapkan firewall untuk melindungi aplikasi dari serangan luar

4. Kesimpulan

Optimalisai dengan penerapan OWASP ZAP merupakan alat yang sangat berguna dalam mengidentifikasi dan menilai kerentanan keamanan pada website. Temuan kerentanan seperti A01, A03, A04, A05, A06, A08, dan A09 yang mencakup ancaman seperti Cross-Site Scripting (XSS), Clickjacking, dan Man-in-the-Middle menyoroti pentingnya penerapan langkah-langkah mitigasi untuk melindungi keamanan situs web. Penerapan solusi seperti konfigurasi header keamanan (CSP, HSTS, dan X-Frame Options) serta perlindungan terhadap data sensitif sangat penting untuk mencegah eksploitasi. Selain itu, penggunaan OWASP ZAP tidak hanya berfungsi untuk mendeteksi dan memperbaiki ancaman, tetapi juga meningkatkan kesadaran tentang perlunya kebijakan keamanan yang tangguh dalam aplikasi web. Oleh karena itu, pemilik website dapat lebih baik melindungi data sensitif, mempertahankan kepercayaan pengguna, dan memastikan keamanan di tengah kompleksitas lingkungan digital saat ini.

Daftar Pustaka

[1] A. F. Hasibuan and D. Handoko, “Analisis Keretanan Website Dengan Aplikasi Owasp Zap,” *J. Ilmu Komput. dan Sist. Inf.*, vol. 2, no. 2, pp. 257–270, 2023, [Online]. Available: https://jurnal.unity-academy.sch.id/index.php/jirsi/article/view/51

[2] L. Kestina, Yuhandri, and G. W. Nurcahyo, “Penanganan Celah Keamanan Website dengan Ethical Hacking dan Issaf Menggunakan Acunetix Vulnerability (Studi Kasus di Bkpsdmd Kabupaten Kerinci),” *Innov. J. Soc. Sci. Res.*, vol. 3, no. 4, pp. 9192–9203, 2023.

[3] M. Trada, J. Teknik, and E. Polbitrada, “Analisis Metode OWASP V4 . 2 dalam Pengujian Keamanan Sistem Informasi Rumah Sakit,” *J. Tek. Elektromedik Polbitrada*, vol. 5, no. 2, pp. 87–97, 2024.

[4] M. I. A. Elfatiha, I. R. Riadi, and R. U. Umar, “Security Analysis of Web-Based Academic Information System using OWASP Framework,” *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, no. 4, 2024, doi:

10.22219/kinetik.v9i4.2015.
[5] S. A. Febriani, A. Muni, B. Rianto, M. Jalil, and Chrismondari, "Analisis Kerentanan Keamanan Sistem Informasi Akademik Menggunakan Owasp-Zap Di Universitas Islam Indragiri," *J. Sist. Inf.*, vol. 2, no. 6, pp. 409–420, 2024.
[6] R. Febriana, "Blackbox Testing Sistem Informasi Absensi Pegawai Karawang Dengan Metode Top 10 Owasp Attack," *J. Ilm. Wahana Pendidik.*, vol. 8, no. 12, pp. 327–334, 2022, [Online]. Available: 10.5281/zenodo.6945632
[7] M. Annas, R. T. Adek, and Y. Afrillia, "Web Application Firewall (WAF) Design to Detect and Anticipate Hacking in Web-Based Applications," *J. Adv. Comput. Knowl. Algorithms*, vol. 1, no. 3, pp. 52–60, 2024, doi: 10.29103/jacka.v1i3.16315.
[8] S. Schmeelk and L. Tao, "A Case Study of Mobile Health Applications: The OWASP Risk of Insufficient Cryptography," *J. Comput. Sci. Res.*, vol. 4, no. 1, pp. 22–31, 2022, doi: 10.30564/jcsr.v4i1.4271.
[9] Y. W, Rodianto, and Zulkarnaen, "Penerapan Framework Cobit 5 Dalam Analisis Keamanan Website Desa Uma Beringin Dengan Metode Capability Maturity Model Integration (Cmmi)," *J. Inform. Teknol. dan Sains*, vol. 4, no. 2, pp. 66–71, 2022, doi: 10.51401/jinteks.v4i2.1569.
[10] A. F. Sebrina, A. Junaidi, and A. N. Sihananto, "Testing posketanmu website with google penetration testing and OWASP Top 10," *J. Mantik*, vol. 8, no. 1, pp. 636–645, 2024, doi: 10.35335/mantik.v8i1.5204.
[11] D. Priyawati, S. Rokhmah, and I. C. Utomo, "Website Vulnerability Testing and Analysis of Internet Management Information System Using OWASP," *Int. J. Comput. Inf. Syst. Peer Rev. J.*, vol. 3, no. 3, pp. 143–147, 2022, doi: 10.29040/ijcis.v3i3.90.
[12] I. O. Riandhanu, "Analisis Metode Open Web Application Security Project (OWASP) Menggunakan Penetration Testing pada Keamanan Website Absensi," *J. Inf. dan Teknol.*, vol. 4, no. 3, pp. 160–165, 2022, doi: 10.37034/jidt.v4i3.236.
[13] Reza. Aditama; Edi. Negara, "Pemindai Kerentanan Terhadap Website Jago Masak Dengan Metode Pengujian Penetrasi OWASP ZAP," *J. Mantik*, vol. 6, no. 3, pp. 3406–3412, 2022.
[14] K. Nisa, M. A. Putra, R. A. Siregar, and M. Dedi Irawan, "Analisis Website Tapanuli Tengah Menggunakan Metode Open Web Application Security Project Zap (Owasp Zap)," *Bull. Inf. Technol.*, vol. 3, no. 4, pp. 308–216, 2022, doi: 10.47065/bit.v3i4.389.
[15] H. Saputra, A. Z. Abidin, F. Faldi, and M. T. Sumadi, "Penetration Testing on Mail Server Website using the OWASP Method," *J. Mandiri IT*, vol. 12, no. 2, pp. 58–65, 2023, [Online]. Available: https://ejournal.isha.or.id/index.php/Mandiri/article/view/232
[16] Y. Todankar, R. Mhatre, O. Dhumal, and A. Patil, "Open Worldwide Application Security Project (OWASP) Operating Systems," *Int. J. Adv. Res. Sci. Commun. Technol.*, vol. 3, no. 2, pp. 150–155, 2024, [Online]. Available: https://owasp.org/about/
[17] M. Idris, I. Syarif, and I. Winarno, "Web Application Security Education Platform Based on OWASP API Security Project," *Emit. Int. J. Eng. Technol.*, vol. 10, no. 2, pp. 246–261, 2022, doi: 10.24003/emitter.v10i2.705.
[18] M. Angelini, S. Bonomi, and A. Palma, "A Methodology to Support Automatic Cyber Risk Assessment Review," *Migr. Lett.*, vol. 5, no. 3, pp. 1–16, 2022, [Online]. Available: http://arxiv.org/abs/2207.03269
[19] E. S. Mena, "Analysis of Risks and Vulnerabilities in a University-Level LMS System Analysis of Risks and Vulnerabilities in a University-Level LMS System," *Migr. Lett.*, vol. 20, no. 11, pp. 1252–1262, 2023, doi: 10.59670/ml.v20iS8.5090.
[20] I. M. Edy Listartha, I. M. A. Premana Mitha, M. W. Aditya Arta, and I. K. W. Yuda Arimika, "Analisis Kerentanan Website SMA Negeri 2 Amlapura Menggunakan Metode OWASP (Open Web Application Security Project)," *Simkom*, vol. 7, no. 1, pp. 23–27, 2022, doi: 10.51717/simkom.v7i1.63.
[21] K. Abdulghaffar, N. Elmrabit, and M. Yousefi, "Enhancing Web Application Security through Automated Penetration Testing with Multiple Vulnerability Scanners," *Computers*, vol. 12, no. 11, pp. 1–17, 2023, doi: 10.3390/computers12110235.

[22] A. W. Kuncoro and F. Rahma, “Analisis Metode Open Web Application Security Project (OWASP) pada Pengujian Keamanan Website: Literature Review,” *Automata*, vol. 3, no. 1, pp. 1–5, 2021, [Online]. Available: https://www.sciencedirect.com