Analisis Metode OWASP V4.2 dalam Pengujian Keamanan Sistem Informasi Rumah Sakit

Analysis of the OWASP V4.2 Method in Hospital Information System Security Testing

Bajeng Nurul Widyaningrum[1], Destri Maya Rani[2], dan Lingga Kurnia Ramadhani[3]
[1)] Rekam Medis dan Informasi Kesehatan Polbitrada, Jl. Sambiroto Raya No.64-D, Sambiroto, Kec. Tembalang, Kota Semarang, Indonesia 50276
[2)] Rekam Medis dan Informasi Kesehatan Polbitrada, Jl. Sambiroto Raya No.64-D, Sambiroto, Kec. Tembalang, Kota Semarang, Indonesia 50276
[3)] Bisnis Digital, Fakultas Sains dan Teknologi, Universitas IVET Semarang, Jl. Pawiyatan Luhur IV No.16, Bendan Duwur, Kec. Gajahmungkur, Kota Semarang, Jawa Tengah 50235
Alamat korespondensi: bnwidyani@gmail.com

Abstrak
Penelitian ini bertujuan untuk mengidentifikasi dan memitigasi kerentanan keamanan pada Sistem Informasi Rumah Sakit (SIMRS) menggunakan metode pengujian berbasis OWASP Web Security Testing Guide (WSTG) v4.2. Dengan bantuan alat OWASP ZAP, berbagai kerentanan berhasil diidentifikasi, seperti SQL Injection, kelemahan dalam manajemen sesi, ketiadaan atribut keamanan pada cookie, dan pengungkapan informasi sensitif melalui URL atau komentar kode. SQL Injection teridentifikasi sebagai kerentanan dengan risiko tertinggi, karena berpotensi memungkinkan pelaku serangan untuk mengakses, memanipulasi, atau menghapus data sensitif dalam basis data. Selain itu, kelemahan pada atribut cookie, seperti HttpOnly dan SameSite, serta ketiadaan mekanisme anti-CSRF, mengindikasikan potensi ancaman berupa Cross-Site Scripting (XSS) dan Cross-Site Request Forgery (CSRF). Penerapan solusi berdasarkan WSTG v4.2 melibatkan langkah-langkah seperti implementasi enkripsi HTTPS, penggunaan prepared statements untuk interaksi basis data, penerapan header keamanan seperti Content-Security-Policy (CSP), dan validasi input untuk mengurangi risiko XSS. Selain itu, audit kode dilakukan untuk menghapus komentar sensitif, sementara file tersembunyi atau cadangan yang tidak diperlukan dihapus untuk meminimalkan potensi kebocoran informasi. Hasil pengujian setelah implementasi solusi menunjukkan peningkatan signifikan dalam tingkat keamanan aplikasi. Penelitian ini membuktikan bahwa pendekatan berbasis WSTG v4.2 dapat memberikan panduan yang komprehensif dan sistematis dalam pengujian keamanan aplikasi web. Dengan hasil ini, organisasi, khususnya di sektor kesehatan, dapat memastikan perlindungan data pasien yang lebih baik dan mematuhi standar keamanan informasi yang berlaku.

Kata kunci: OWASP 4.2, Pengujian Penetrasi, Keamanan Website, Sistem Informasi Rumah Sakit, Keamanan Siber

Abstract
This research aims to identify and mitigate security vulnerabilities in the Hospital Information System (SIMRS) using the OWASP Web Security Testing Guide (WSTG) v4.2 based testing method. With the help of the OWASP ZAP tool, various vulnerabilities were identified, such as SQL Injection, weaknesses in session management, lack of security attributes in cookies, and disclosure of sensitive information through URLs or code comments. SQL Injection was identified as the highest risk vulnerability, as it potentially allows attackers to access, manipulate, or delete sensitive data in the database. In addition, weaknesses in cookie attributes, such as HttpOnly and SameSite, and the absence of an anti-CSRF mechanism, indicate potential threats in the form of Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF). The implementation of a solution based on WSTG v4.2 involves steps such as the implementation of HTTPS encryption, the use of prepared statements for database interaction, the application of security headers such as Content-Security-Policy (CSP), and input validation to reduce the risk of XSS. In addition, code audits were conducted to remove sensitive comments, while hidden files or unnecessary backups were removed to minimize the potential for information leakage. Test results after the implementation of the solution showed a significant improvement in the security level of the application. This research proves that the WSTG v4.2-

based approach can provide comprehensive and systematic guidance in web application security testing. With these results, organizations, particularly in the healthcare sector, can ensure better protection of patient data and comply with applicable information security standards.

Keywords: OWASP 4.2, Penetration Testing, Website Security, Hospital Information System, Cyber Security

Pendahuluan

Seiring perkembangan teknologi yang semakin pesat, hampir semua bidang kehidupan menggunakan teknologi untuk membuat hidup lebih mudah bagi orang-orang untuk melakukan hal-hal tertentu, terutama teknologi internet. Dengan demikian, pengembang aplikasi berbasis web profesional dan orang-orang yang berkecimpung di dunia teknologi berlomba-lomba untuk membuat aplikasi baru yang dapat digunakan dalam kehidupan sehari-hari (1).

Bidang pelayanan kesehatan contohnya. Pada bidang pelayanan kesehatan telah terpengaruh oleh perkembangan teknologi informasi yang begitu pesat. Teknologi informasi memungkinkan pengiriman data kesehatan secara mudah dan cepat, memproses dan mengolahnya menjadi informasi, dan menyimpan lebih banyak data. Dengan demikian, teknologi informasi membantu manajemen rumah sakit menjadi lebih efisien (2). Sistem Informasi Rumah Sakit berbasis web ini juga memanfaatkan media informasi untuk kemajuan teknologi dan informasi, yang meningkatkan kualitas pelayanan, waktu yang dihabiskan, dan biaya di fasilitas pelayanan kesehatan (3).

Salah satu kelemahan sistem dapat terletak pada kemudahan pertukaran dan pengelolaan informasi yang dilakukan secara online. Karena cara website bekerja, data yang disimpan di sana dapat berupa informasi pribadi atau rahasia. Dalam membangun suatu website, keamanan dan perlindungan diperlukan untuk memastikan bahwa data pribadi tidak jatuh ke tangan yang salah. Pengujian keamanan website juga dikenal sebagai metode penetrasi testing yaitu melakukan simulasi serangan digital terhadap website yang bersangkutan (4)(5).

Tujuan pengujian penetrasi web adalah untuk menemukan gangguan keamanan pada situs web yang dapat dikategorikan sebagai kerentanan keamanan (6). Tahapan penetrasi web dilakukan pada hasil dan melalui berbagai modul yang disesuaikan dengan framework atau standar yang tersedia. Agar hasil pengujian bersifat valid dan dapat dipertanggung jawabkan, penguji harus menggunakan framework. ISSAF dan OWASP Testing Guide adalah dua framework yang dapat digunakan untuk penetration testing (7).

Parameter keamanan yang dikembangkan oleh OWASP adalah salah satu metode yang dapat digunakan untuk melakukan penetrasi tes website (8). OWASP (Open Web Application Security Project) adalah kerangka kerja yang menawarkan pedoman untuk keamanan perangkat lunak, terutama untuk website yang memberikan keamanan sistem melalui proyek open-source, serta tools yang didukung OWASP untuk pengujian sistem (9).

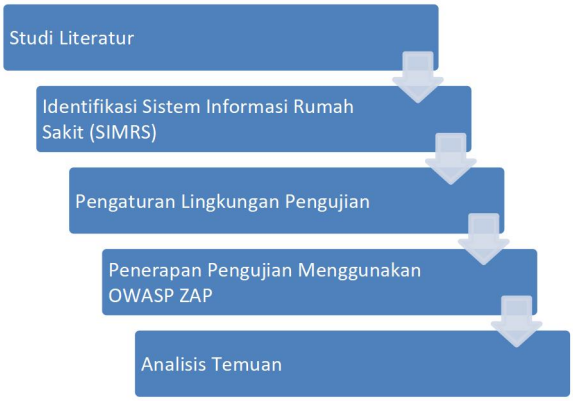
Sebagai panduan kerangka kerja yang dirilis oleh OWASP, Web Security Testing Guide (WSTG) menjelaskan langkah-langkah yang harus dilakukan untuk melakukan analisis keamanan pada sistem berbasis web. WSG adalah panduan lengkap untuk pengujian keamanan aplikasi dan layanan web. Setiap tahun, WSTG mengeluarkan versi baru, WSTG V4.2 (10). Metode OWASP versi 4 memungkinkan analisis kerentanan aplikasi berbasis web untuk menilai keamanan suatu aplikasi berdasarkan hasil pengujian kerentanan pada website menggunakan beberapa tahapan kategori, yaitu tahap Authentication Testing Authorization, Session Management Testing, Input Validation Testing, dan Error Handling. Metode ini dapat digunakan sebagai standar penilaian keamanan aplikasi berbasis web (11).

Tujuan dari penelitian ini adalah untuk menganalisis dan melakukan penelitian tentang penggunaan metode OWASP dalam pengujian keamanan sistem informasi rumah sakit pada laboratorium Polbitrada. Diharapkan bahwa penelitian ini akan memberikan gambaran tentang pengujian keamanan sistem, khususnya pengujian keamanan website menggunakan metode OWASP, seperti yang telah dilakukan oleh penelitian sebelumnya.

Metode

a. Gambaran Umum Penelitian

Metode penelitian untuk pengujian keamanan Sistem Informasi Rumah Sakit (SIMRS) menggunakan OWASP Web Security Testing Guide (WSTG) v4.2 dan alat OWASP ZAP (Zed Attack Proxy) dirancang secara sistematis untuk mengidentifikasi kerentanan dan memberikan solusi perbaikan (12)(13). Penelitian ini dimulai dengan studi literatur untuk memahami standar keamanan aplikasi web berdasarkan WSTG v4.2 dan fitur OWASP ZAP.



Gambar 1. Framework Metodologi Penelitian

b. Studi Literatur

Studi literatur dilakukan untuk memahami keamanan aplikasi web berdasarkan WSTG v4.2 dan OWASP ZAP. Aktivitas ini mencakup kajian dokumen panduan, jurnal, dan laporan keamanan, khususnya di sektor kesehatan. Hasil dari tahap ini adalah kerangka kerja awal untuk pengujian keamanan SIMRS.

c. Identifikasi Sistem Informasi Rumah Sakit (SIMRS)

Tahap ini bertujuan memahami arsitektur dan modul SIMRS. Kegiatan meliputi wawancara dengan pengembang, analisis teknologi yang digunakan, dan identifikasi modul penting seperti login dan manajemen data pasien. Hasilnya adalah peta arsitektur dan cakupan pengujian.

d. Pengaturan Lingkungan Pengujian

Lingkungan pengujian disiapkan dengan menginstal dan mengonfigurasi OWASP ZAP. SIMRS ditempatkan pada server lokal atau sistem staging untuk memastikan keamanan saat pengujian. Hasil dari tahap ini adalah lingkungan siap pakai untuk pengujian.

e. Penerapan Pengujian dengan OWASP ZAP

Pengujian dilakukan sesuai kategori WSTG v4.2 menggunakan OWASP ZAP (14). Ini mencakup pengumpulan informasi, pengujian konfigurasi, validasi input, pengujian autentikasi, an analisis manajemen sesi. Setiap pengujian menghasilkan temuan kerentanan teknis yang relevan.

f. Analisis Temuan

Temuan kerentanan dianalisis dan diklasifikasikan berdasarkan tingkat risiko. Analisis difokuskan pada dampaknya terhadap keamanan data pasien dan operasional rumah sakit. Hasil dari tahap ini adalah daftar risiko yang terprioritaskan.

g. Rekomendasi Perbaikan

Rencana perbaikan disusun untuk setiap kerentanan, mencakup langkah seperti penerapan header keamanan, penguatan autentikasi, dan validasi input. Dokumen rekomendasi ini dirancang agar implementasi dapat dilakukan dengan mudah oleh tim pengembang.

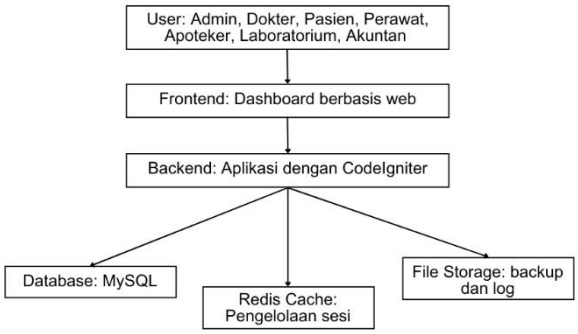
h. Kesimpulan

Hasil disusun untuk mencatat seluruh proses, temuan, dan perbaikan. hasil ini mencakup dokumen teknis untuk pengembang dan laporan ringkas untuk manajemen rumah sakit, yang digunakan sebagai panduan pengembangan berkelanjutan.

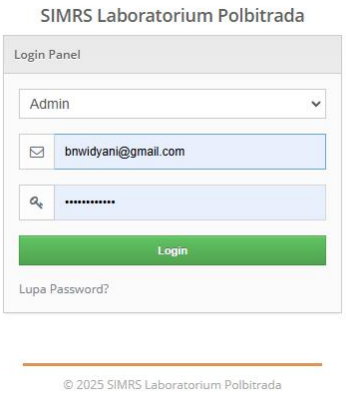
Hasil

a. Identifikasi Sistem Informasi Rumah Sakit (SIMRS)

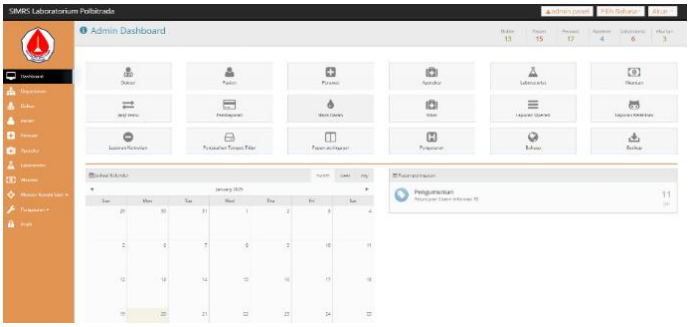
Sistem Informasi Rumah Sakit (SIMRS) Polbitrada adalah platform digital yang dirancang untuk mengelola berbagai aktivitas operasional rumah sakit. Berdasarkan dashboard yang dianalisis, SIMRS memiliki antarmuka yang terstruktur dengan berbagai modul seperti manajemen dokter, pasien, perawat, apoteker, laboratorium, dan akuntansi. Fitur seperti penjadwalan janji temu, laporan medis, manajemen tempat tidur, dan stok bank darah membantu dalam pengelolaan layanan rumah sakit secara efisien. Berikut gambaran dari arsitektur SIMRS:



Gambar 2. Arsitektur SIMRS



Gambar 3. Tampilan Login SIMRS Polbitrada



Gambar 4. Dashboard SIMRS Polbitrada

Dari perspektif teknis, SIMRS dibangun menggunakan framework CodeIgniter untuk backend, yang merupakan framework PHP ringan dan efisien. Antarmuka menggunakan HTML, CSS, dan JavaScript, kemungkinan dengan bantuan framework frontend seperti Bootstrap. Sistem ini dirancang untuk mendukung berbagai peran pengguna, seperti admin, dokter, dan perawat, dengan menyediakan akses sesuai peran mereka. Namun, seperti sistem informasi lainnya, SIMRS harus memastikan keamanan, khususnya terhadap ancaman seperti SQL Injection, Cross-Site Scripting (XSS), dan manajemen sesi yang tidak aman. Upaya pengamanan ini dapat diperkuat dengan menerapkan standar OWASP WSTG dan alat seperti OWASP ZAP.

Tabel 1. Sistem Informasi Rumah Sakit (SIMRS)

Aspek	Analisis
Modul Utama	Dokter, Pasien, Perawat, Apoteker, Laboratorium, Akuntansi, Manajemen Tempat Tidur.
Fitur Pendukung	Janji Temu, Laporan Medis (Operasi, Kelahiran, Kematian), Bank Darah, Papan Peringatan.
Antarmuka Pengguna	Terstruktur, mudah digunakan, dan menyediakan navigasi yang jelas untuk pengguna.
Teknologi yang Digunakan	Backend: CodeIgniter (PHP Framework). Frontend: HTML, CSS, JavaScript (Bootstrap).
Keunggulan	Terintegrasi untuk berbagai entitas rumah sakit, pengelolaan operasional yang efisien.
Peran Pengguna	Admin, Dokter, Perawat, Akuntan, Apoteker, Pasien.

Cakupan pengujian yang dilakukan untuk sistem informasi mencakup berbagai aspek seperti fitur, metode, lingkungan pengujian, hasil yang diharapkan, dan hasil aktual.

Tabel 2. Cakupan Pengujian Sistem Informasi

ID Pengujian	Fitur yang Diuji	Metode Pengujian	Lingkungan	Hasil yang Diharapkan	Hasil Aktual
1	Login pengguna	Black-box testing	Windows, Chrome 133.0.6943.16	Login berhasil	Login berhasil
2	CRUD data pengguna	Black-box testing	Linux, Firefox 115	Data tersimpan di database	Data tersimpan
3	Integrasi API	White-box testing	Ubuntu Server 20.04	Respon API sesuai dokumentasi	Respon sesuai
4	Load testing	Load testing	Windows Server 2019	Menangani 1000 request/detik	Berhasil

b. Pengaturan Lingkungan Pengujian

1. Sistem Operasi dan Perangkat Pengujian

Pengujian dilakukan pada perangkat dengan sistem operasi Windows 11 Pro versi 23H2, yang memiliki arsitektur 64-bit. Spesifikasi perangkat mencakup prosesor AMD Ryzen 5 2600 Six-Core Processor dengan kecepatan 3.40 GHz, RAM sebesar 8 GB, dan build OS 22631.4751. Dengan konfigurasi ini, perangkat dinilai memadai untuk menjalankan proses pengujian menggunakan OWASP ZAP yang membutuhkan pemrosesan intensif. Selain itu, perangkat ini mendukung Windows Feature Experience Pack versi 1000.22700.1055.0, yang memastikan kompatibilitas dengan aplikasi terbaru.

2. Alat Pengujian

Versi OWASP ZAP 2.16.0 digunakan

sebagai alat utama dalam proses pengujian. Versi ini merupakan rilis stabil dengan fitur-fitur seperti Passive Scan, Active Scan, serta integrasi dengan browser modern untuk mendukung pengujian keamanan aplikasi web. ZAP versi 2.16.0 kompatibel dengan Windows 11 Pro, memastikan pengujian dapat dilakukan tanpa kendala kompatibilitas perangkat lunak.

3. Persiapan dan Konfigurasi Lingkungan

Untuk menciptakan lingkungan pengujian yang optimal, langkah-langkah berikut dilakukan:

a. Instalasi dan Konfigurasi OWASP ZAP: OWASP ZAP diinstal dan dijalankan dengan konfigurasi proxy server pada alamat IP 127.0.0.1 dan port 8080 (default). Browser pengujian dikonfigurasi untuk terhubung ke proxy ZAP, dan sertifikat keamanan ZAP diimpor ke browser untuk memungkinkan analisis lalu lintas HTTPS.

b. Konfigurasi Sistem Operasi: Sistem operasi diperbarui ke versi terbaru untuk memastikan patch keamanan terkini telah diterapkan. Firewall dan antivirus sementara dinonaktifkan apabila memblokir lalu lintas yang diperlukan untuk pengujian. Stabilitas koneksi jaringan lokal diverifikasi, terutama jika pengujian melibatkan server staging atau sistem lokal.

c. Lingkungan Aplikasi yang Diuji: Sistem Informasi Rumah Sakit (SIMRS) diuji pada server staging untuk menghindari dampak pada sistem produksi. Koneksi ke SIMRS dilakukan melalui jaringan lokal atau IP publik yang telah dikonfigurasi.

d. Optimalisasi Performa OWASP ZAP: Mengingat perangkat memiliki RAM sebesar 8 GB, alokasi memori untuk OWASP ZAP disesuaikan dengan menambahkan argumen JVM seperti Xmx4g. Argumen ini memastikan ZAP dapat menangani pemrosesan data dalam jumlah besar tanpa hambatan performa.

4. Validasi Lingkungan Pengujian

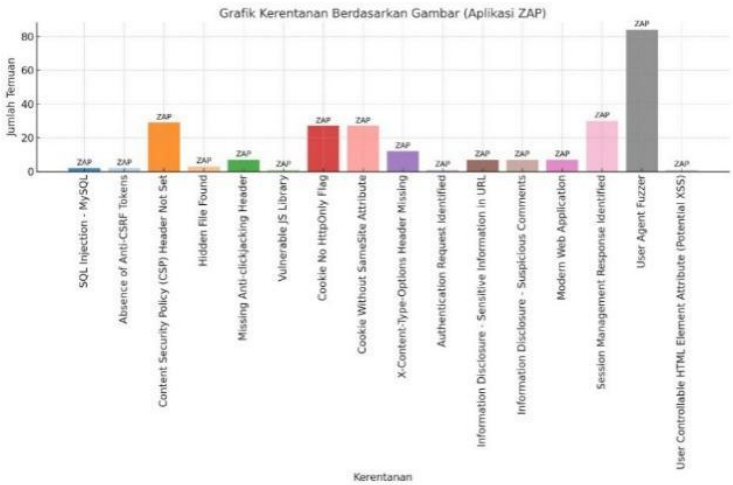
Lingkungan pengujian divalidasi dengan melakukan Passive Scan awal untuk memastikan konfigurasi proxy dan koneksi ke aplikasi SIMRS berjalan tanpa kendala. Uji coba koneksi browser dengan ZAP dilakukan untuk memastikan sertifikat keamanan ZAP diterima, memungkinkan analisis penuh terhadap lalu lintas HTTP/HTTPS.

Tabel 3. Pengaturan Lingkungan Pengujian

Aspek	Deskripsi
Sistem Operasi	Windows 11 Pro, versi 23H2, arsitektur 64-bit
Spesifikasi Perangkat	Prosesor AMD Ryzen 5 2600, 3.40 GHz; RAM 8 GB; Build OS 22631.4751
Alat Pengujian	OWASP ZAP versi 2.16.0
Instalasi dan Konfigurasi	- Proxy server: IP 127.0.0.1, port 8080. - Sertifikat keamanan ZAP diimpor ke browser.
Konfigurasi Sistem Operasi	- Sistem diperbarui ke versi terbaru. - Firewall dan antivirus dinonaktifkan sementara jika memblokir lalu lintas ZAP.
Lingkungan Aplikasi yang Diuji	- SIMRS diuji pada server staging. - Koneksi melalui jaringan lokal atau IP publik.
Optimalisasi Performa ZAP	Menambahkan argumen JVM: Xmx4g untuk alokasi memori maksimum 4 GB.
Validasi Lingkungan	- Passive scan dilakukan untuk memvalidasi konfigurasi. - Browser diuji untuk memastikan sertifikat ZAP diterima.

c. Penerapan Pengujian dengan OWASP ZAP

Grafik pada Gambar 5 menunjukkan distribusi jumlah temuan kerentanan pada sistem informasi rumah sakit (SIMRS) berdasarkan tools ZAP. Setiap jenis kerentanan ditampilkan bersama jumlah temuan untuk memprioritaskan tindakan mitigasi.



Gambar 5. Grafik Kerentanan SIMRS dengan Tools ZAP

Tabel 4: Kerentanan SIMRS dengan Tools ZAP

Kerentanan	Jumlah Temuan
SQL Injection - MySQL	2
Absence of Anti-CSRF Tokens	2
Content Security Policy (CSP) Header Not Set	29
Hidden File Found	3
Missing Anti-clickjacking Header	7
Vulnerable JS Library	1
Cookie No HttpOnly Flag	27
Cookie Without SameSite Attribute	27
X-Content-Type-Options Header Missing	12
Authentication Request Identified	1
Information Disclosure - Sensitive Information in URL	7
Information Disclosure - Suspicious Comments	7
Modern Web Application	7
Session Management Response Identified	30
User Agent Fuzzer	84
User Controllable HTML Element Attribute (Potential XSS)	1

Hasil analisis keamanan aplikasi berdasarkan OWASP Web Security Testing Guide (WSTG) v4.2 mengidentifikasi beberapa kerentanan kritis yang memerlukan mitigasi untuk mencegah potensi eksploitasi. Salah satu kerentanan dengan tingkat risiko tertinggi adalah SQL Injection (WSTG-INPV-05), yang memungkinkan pelaku menyuntikkan perintah berbahaya untuk mengakses atau memanipulasi basis data. Selain itu, ketiadaan token anti-CSRF, seperti yang ditunjukkan pada Absence of Anti-CSRF Tokens (WSTG-SESS-05), meningkatkan risiko serangan Cross-Site Request Forgery (CSRF), yang dapat mengakibatkan manipulasi tidak sah atas tindakan pengguna.

Kerentanan lain, seperti Hidden File Found (WSTG-CONF-04), menunjukkan adanya file tersembunyi yang berpotensi mengungkap informasi sensitif jika tidak ditangani. Serangan clickjacking juga menjadi perhatian utama, dengan kerentanan Missing Anti-clickjacking Header (WSTG-CLNT-09), di mana pengguna dapat diarahkan untuk berinteraksi dengan antarmuka aplikasi tanpa disadari. Atribut cookie yang tidak diatur dengan benar, seperti ketiadaan HttpOnly dan SameSite, yang teridentifikasi dalam Cookie No HttpOnly Flag dan Cookie Without SameSite Attribute (WSTG-SESS-02), memperbesar risiko serangan berbasis XSS dan CSRF.Kerentanan lainnya termasuk Authentication Request Identified (WSTG-ATHN-01), yang menunjukkan kebutuhan untuk memastikan kredensial pengguna ditransmisikan melalui saluran yang terenkripsi, serta Information Disclosure dalam URL dan komentar kode sumber, yang dapat memberikan informasi sensitif kepada pelaku. Selain itu, kerentanan seperti Session Management Response Identified (WSTG-SESS-01) menunjukkan kelemahan dalam pengelolaan sesi, sedangkan User Agent Fuzzer (WSTG-INFO-02) dapat digunakan untuk mengeksploitasi informasi server.

Tabel 5. hasil (WSTG) versi 4.2

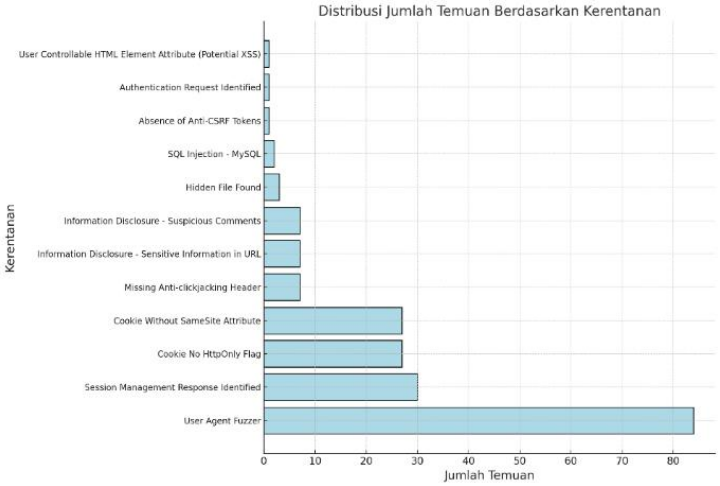
No	Kerentanan	WSTG V4.2
1	SQL Injection	WSTG-INPV-05
2	Absence of Anti-CSRF Tokens	WSTG-SESS-05
3	Hidden File Found	WSTG-CONF-04
4	Missing Anti-clickjacking Header	WSTG-CLNT-09
5	Cookie No HttpOnly Flag	WSTG-SESS-02
6	Cookie Without SameSite Attribute	WSTG-SESS-02
7	Authentication Request Identified	WSTG-ATHN-01
8	Information Disclosure - Sensitive Information in URL	WSTG-INFO-05
9	Information Disclosure - Suspicious Comments	WSTG-INFO-05
10	Session Management Response Identified	WSTG-SESS-01
11	User Agent Fuzzer	WSTG-INFO-02
12	User Controllable HTML Element Attribute (Potential XSS)	WSTG-INPV-01

d. Analisis Temuan

Pengujian keamanan aplikasi web yang dilakukan berdasarkan panduan OWASP Web Security Testing Guide (WSTG) v4.2 berhasil mengidentifikasi sejumlah kerentanan dengan dampak yang beragam terhadap sistem. Salah satu temuan utama adalah User Agent Fuzzer, yang muncul sebanyak 84 kali dan mengindikasikan adanya risiko eksploitasi informasi server melalui manipulasi header User-Agent. Kerentanan lain yang signifikan melibatkan kelemahan manajemen sesi, seperti Session Management Response Identified, serta pengaturan atribut cookie, yaitu Cookie No HttpOnly Flag dan Cookie Without SameSite Attribute. Kedua kerentanan tersebut dapat dimanfaatkan untuk serangan serius seperti pembajakan sesi (session hijacking) dan Cross-Site Request Forgery (CSRF).

Kerentanan yang terkait dengan injeksi

SQL (SQL Injection) meskipun jumlah temuan relatif sedikit, memiliki tingkat risiko sangat tinggi karena dapat memungkinkan pelaku untuk memanipulasi atau mendapatkan akses ilegal ke basis data. Selain itu, kelemahan pada header keamanan seperti tidak diterapkannya anti-clickjacking header memperlihatkan potensi eksploitasi antarmuka pengguna. Sementara itu, Hidden File Found yang ditemukan menunjukkan risiko rendah hingga sedang, namun masih perlu diperhatikan jika berisi informasi sensitif.



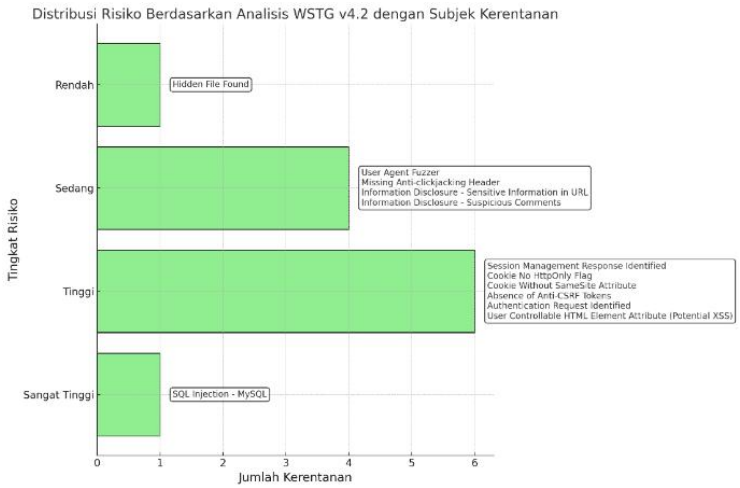
Gambar 6. Grafik (WSTG) versi 4.2 Berdasarkan Jumlah Temuan

Tabel 6. Hasil (WSTG) versi 4.2 Berdasarkan Resiko

No	Kerentanan	WSTG V4.2	Jumlah Temuan	Risiko
1	User Agent Fuzzer	WSTG-INFO-02	84	Potensi eksploitasi informasi server (seperti jenis server atau framework yang digunakan) oleh penyerang untuk mempersiapkan serangan lebih lanjut seperti brute force atau serangan berbasis kerentanan spesifik server.
2	Session Management Response Identified	WSTG-SESS-01	30	Kerentanan dalam manajemen sesi dapat memungkinkan pembajakan sesi (*session hijacking*), pengambilan alih akun, atau penyalahgunaan sesi autentikasi.
3	Cookie No HttpOnly Flag	WSTG-SESS-02	27	Cookie dapat diakses oleh skrip pihak ketiga melalui JavaScript, memungkinkan serangan seperti **Cross-Site Scripting (XSS)** atau pencurian sesi.
4	Cookie Without SameSite Attribute	WSTG-SESS-02	27	Cookie dapat digunakan lintas situs tanpa batasan, memungkinkan serangan **Cross-Site Request Forgery (CSRF)** atau penyalahgunaan autentikasi.
5	Missing Anti-clickjacking Header	WSTG-CLNT-09	7	Aplikasi dapat menjadi target serangan clickjacking, di mana penyerang mengelabui pengguna untuk berinteraksi dengan elemen sensitif tanpa sepengetahuan mereka.
6	Information Disclosure - Sensitive Information in URL	WSTG-INFO-05	7	Informasi sensitif dalam URL dapat diakses oleh pihak ketiga melalui log server atau cache browser, meningkatkan risiko kebocoran data.
7	Information Disclosure - Suspicious Comments	WSTG-INFO-05	7	Komentar dalam kode sumber dapat mengungkap informasi sensitif seperti jalur file, API key, atau kredensial, yang dapat digunakan oleh penyerang.
8	Hidden File Found	WSTG-CONF-04	3	File tersembunyi atau cadangan dapat mengandung informasi sensitif atau konfigurasi yang dapat dieksploitasi oleh penyerang.
9	SQL Injection - MySQL	WSTG-INPV-05	2	Penyerang dapat menyisipkan perintah SQL berbahaya untuk mengambil, memodifikasi, atau menghapus data sensitif dalam database.
10	Absence of Anti-CSRF Tokens	WSTG-SESS-05	2	Aplikasi menjadi rentan terhadap serangan CSRF, di mana tindakan tidak sah dilakukan atas

				nama pengguna yang sah.
11	Authentication Request Identified	WSTG-ATHN-01	1	Jika kredensial dikirim tanpa enkripsi, informasi autentikasi dapat disadap melalui serangan man-in-the-middle.
12	User Controllable HTML Element Attribute (Potential XSS)	WSTG-INPV-01	1	Penyerang dapat menyisipkan skrip berbahaya dalam atribut HTML yang dieksekusi oleh browser, memungkinkan pencurian data atau manipulasi antarmuka.

Hasil analisis ini menekankan pentingnya penerapan pengamanan berbasis header, seperti Content Security Policy (CSP), pembaruan eksternal untuk meminimalkan risiko dari serangan, dan penggunaan enkripsi HTTPS untuk melindungi data autentikasi. Dengan pendekatan sistematis seperti yang dijelaskan dalam WSTG v4.2, pengujian keamanan dapat memberikan hasil yang komprehensif, memungkinkan organisasi untuk mengidentifikasi dan mengatasi potensi kerentanan sebelum dieksploitasi oleh pihak tidak bertanggung jawab.



Gambar 7. Distribusi Resiko (WSTG) versi 4.2

e. Rekomendasi Perbaikan

Implementasi perbaikan berbasis OWASP Web Security Testing Guide (WSTG) v4.2 merupakan langkah strategis untuk meningkatkan keamanan aplikasi web dengan mengatasi berbagai kerentanan yang teridentifikasi. Salah satu fokus utama adalah penguatan pada manajemen sesi, termasuk penggunaan token sesi, pengaturan kedaluwarsa, serta pengamanan transmisi data melalui enkripsi HTTPS. Upaya ini dirancang untuk mencegah eksploitasi autentikasi dan mendorong perlindungan terhadap akses ilegal. Selain itu, kelemahan pada atribut cookie, harus diterapkannya HttpOnly dan SameSite, untuk mencegah akses cookie melalui skrip sisi klien sehingga mengurangi risiko serangan berbasis Cross-Site Scripting (XSS) dan Cross-Site Request Forgery (CSRF), yang memiliki dampak signifikan terhadap integritas dan kerahasiaan data.

Penggunaan header keamanan seperti X-Frame-Options dan Content Security Policy (CSP) menjadi langkah mitigasi utama untuk mencegah serangan clickjacking dan pembatasan sumber daya dari domain yang ilegal. Di sisi lain, ancaman injeksi SQL diatasi dengan penerapan prepared statements dan parameterized queries, yang secara efektif mengurangi risiko manipulasi basis data. kemudian, harus dilakukan audit terhadap kode sumber dan penghapusan komentar mencurigakan agar terhindar dari pengungkapan informasi rahasia yang tidak disengaja.

Penerapan validasi input dan output encoding memberikan perlindungan tambahan terhadap potensi serangan XSS, sementara penghapusan file tersembunyi atau cadangan yang tidak diperlukan menutup celah kebocoran informasi konfigurasi. Solusi ini tidak hanya berhasil mengurangi kerentanan secara signifikan tetapi juga memberikan pendekatan yang standar untuk meningkatkan ketahanan aplikasi terhadap ancaman keamanan siber. Dengan mengintegrasikan langkah-langkah ini, aplikasi dapat mencapai tingkat keamanan yang lebih tinggi, mendukung kepercayaan pengguna, dan memastikan kepatuhan terhadap praktik terbaik di industri keamanan informasi.

Tabel 7. Rekomendasi Solusi SIMRS Berdasarkan (WSTG) V4.2

No	Kerentanan	Kategori WSTG	Solusi
1	User Agent Fuzzer	WSTG-INFO-02	Gunakan pengaturan server untuk membatasi informasi sensitif yang diekspos melalui header respons, seperti nama server, versi, dan framework yang digunakan. Terapkan header keamanan seperti X-Content-Type-Options untuk membatasi eksploitasi berbasis header.
2	Session	WSTG-	Perkuat manajemen sesi

	Management Response Identified	SESS-01	dengan menggunakan token sesi yang unik, membatasi waktu kedaluwarsa sesi, dan memastikan bahwa sesi hanya dapat diakses melalui koneksi HTTPS. Audit semua mekanisme pengelolaan sesi untuk mendeteksi kelemahan potensial.
3	Cookie No HttpOnly Flag	WSTG-SESS-02	Terapkan atribut HttpOnly pada cookie autentikasi untuk mencegah akses cookie melalui skrip sisi klien (client-side scripts), sehingga melindungi dari serangan Cross-Site Scripting (XSS).
4	Cookie Without SameSite Attribute	WSTG-SESS-02	Gunakan atribut SameSite pada cookie dengan nilai Strict atau Lax untuk mencegah pengiriman cookie lintas situs, yang dapat digunakan dalam serangan Cross-Site Request Forgery (CSRF).
5	Missing Anti-clickjacking Header	WSTG-CLNT-09	Tambahkan header X-Frame-Options dengan nilai DENY atau SAMEORIGIN untuk melindungi aplikasi dari serangan clickjacking. Alternatif lain adalah menggunakan Content Security Policy (CSP) dengan kebijakan yang melarang iframe pihak ketiga.
6	Information Disclosure - Sensitive Information in URL	WSTG-INFO-05	Hindari penggunaan URL untuk menyimpan atau mentransmisikan informasi sensitif seperti token autentikasi atau kredensial pengguna. Gunakan metode POST untuk pengiriman data sensitif, dan cegah pencatatan URL sensitif di log server.
7	Information Disclosure - Suspicious Comments	WSTG-INFO-05	Audit kode sumber aplikasi untuk menghapus komentar yang berisi informasi sensitif atau rahasia. Terapkan proses tinjauan kode sebelum penerapan ke lingkungan produksi.
8	Hidden File Found	WSTG-CONF-04	Identifikasi dan hapus file tersembunyi atau cadangan yang tidak diperlukan. Pastikan file yang masih diperlukan memiliki izin akses yang ketat untuk mencegah eksploitasi.
9	SQL Injection - MySQL	WSTG-INPV-05	Gunakan parameterized queries dan prepared statements untuk semua interaksi dengan basis data. Terapkan validasi input yang ketat untuk memastikan hanya data yang sesuai yang diterima oleh aplikasi.
10	Absence of Anti-CSRF Tokens	WSTG-SESS-05	Tambahkan token anti-CSRF pada setiap permintaan yang memerlukan autentikasi. Token ini harus unik untuk setiap sesi pengguna dan diverifikasi oleh server untuk memastikan keabsahan permintaan.
11	Authentication Request Identified	WSTG-ATHN-01	Gunakan HTTPS untuk melindungi data autentikasi selama transmisi. Terapkan autentikasi multi-faktor (MFA) untuk meningkatkan keamanan proses autentikasi, dan gunakan enkripsi yang kuat pada data autentikasi.
12	User Controllable HTML Element Attribute (Potential XSS)	WSTG-INPV-01	Terapkan validasi input untuk memastikan bahwa data yang dikirimkan oleh pengguna sesuai dengan format yang diharapkan. Gunakan output encoding pada elemen HTML untuk mencegah eksekusi skrip berbahaya.

Pembahasan

Setelah menerapkan solusi yang diusulkan berdasarkan pedoman OWASP Web Security Testing Guide (WSTG) v4.2, hasil pengujian keamanan aplikasi menggunakan alat OWASP ZAP menunjukkan peningkatan yang signifikan dalam mengurangi kerentanan. Implementasi solusi seperti penambahan atribut HttpOnly dan SameSite pada cookie berhasil mengurangi risiko serangan Cross-Site Request Forgery (CSRF) dan pembajakan sesi. Selain itu, penerapan header keamanan seperti X-Frame-Options dan Content-Security-Policy (CSP) secara efektif mencegah serangan clickjacking dan Cross-Site Scripting (XSS).

Pengelolaan sesi yang lebih aman melalui token sesi unik dan enkripsi HTTPS juga memperkuat mekanisme autentikasi, sehingga mencegah akses tidak sah. Di sisi lain, pembersihan file tersembunyi atau cadangan yang tidak diperlukan berhasil mengurangi risiko kebocoran informasi konfigurasi. Lebih jauh lagi, penggunaan prepared statements dan validasi input dalam pengelolaan basis data secara langsung menghilangkan kerentanan terhadap SQL Injection, yang sebelumnya diidentifikasi sebagai risiko sangat tinggi.

Alat OWASP ZAP, yang digunakan untuk

memvalidasi keamanan aplikasi setelah implementasi, menunjukkan bahwa semua kerentanan utama telah diatasi atau mitigasi secara signifikan, sesuai dengan kategori pengujian WSTG v4.2. Langkah-langkah yang diambil tidak hanya meningkatkan keamanan aplikasi, tetapi juga memastikan kepatuhan terhadap standar keamanan web yang berlaku. Keberhasilan ini menegaskan pentingnya panduan WSTG v4.2 dalam mengidentifikasi dan mengurangi kerentanan, sekaligus menunjukkan efektivitas pendekatan sistematis untuk pengujian keamanan aplikasi web.

Kesimpulan

Penelitian ini berhasil menunjukkan pentingnya penerapan metode pengujian keamanan berbasis OWASP Web Security Testing Guide (WSTG) v4.2 dalam mengidentifikasi dan mengatasi berbagai kerentanan pada Sistem Informasi Rumah Sakit (SIMRS). Dengan menggunakan alat OWASP ZAP, kerentanan seperti SQL Injection, Cross-Site Request Forgery (CSRF), Cross-Site Scripting (XSS), dan manajemen sesi yang tidak aman berhasil diidentifikasi dan dimitigasi melalui langkah-langkah perbaikan yang sesuai dengan kategori pengujian WSTG v4.2. Hasil analisis menunjukkan bahwa kerentanan SQL Injection memiliki dampak paling kritis terhadap keamanan data, sementara kelemahan dalam pengaturan cookie dan manajemen sesi juga menimbulkan risiko signifikan terhadap autentikasi pengguna. Selain itu, ditemukan kerentanan tingkat rendah seperti file tersembunyi yang dapat meningkatkan risiko kebocoran data apabila tidak ditangani. Implementasi solusi berbasis WSTG v4.2, seperti penggunaan enkripsi HTTPS, penguatan validasi input, penerapan header keamanan, dan audit kode sumber, terbukti efektif dalam meningkatkan ketahanan sistem terhadap serangan siber.

Keseluruhan proses pengujian memberikan panduan yang komprehensif untuk meningkatkan keamanan aplikasi berbasis web, khususnya di sektor layanan kesehatan. Penerapan pendekatan berbasis WSTG v4.2 tidak hanya memastikan kepatuhan terhadap standar keamanan informasi yang berlaku tetapi juga memberikan kontribusi signifikan dalam melindungi data pasien yang bersifat sensitif. Keberhasilan penelitian ini dapat menjadi dasar pengembangan pengujian keamanan pada aplikasi lain di berbagai sektor kritis.

Daftar Rujukan

1. Nurulita F, Sofiana S. Perancangan Sistem Informasi Rekam Medis Berbasis Web (Studi Kasus: Klinik CAS Medica). Bul Ilm Ilmu Komput dan … [Internet]. 2023;1(2):227–36. Available from: http://jurnalmahasiswa.com/index.php/biikma/article/view/379%0Ahttp://jurnalmahasiswa.com/index.php/biikma/article/download/379/261
2. Rani P, Chakraborty MK, Sah RPRPRP, Subhashi A, Disna R, UIP P, et al. No Titleالأنا والآخر ودوي زالغرب. Range Manag Agrofor [Internet]. 2020;4(1):1–15. Available from: http://dx.doi.org/10.1016/j.asw.2013.04.001%5Cnhttp://journals.cambridge.org/abstract_S0140525X00005756%5CnLib scanned%5Cnhttp://www.br-ie.org/pub/index.php/rbie/article/view/1293%5Cnhttp://www-psych.nmsu.edu/~pfoltz/reprints/Edmedia99.html%5Cnhttp://urd.
3. Prabowo DWS, Triono J. Rancang Bangun Sistem Informasi Konsultasi Medis Berbasis Web. Pilar Teknol. 2021;6(1):8–14.
4. Dharmawangsa IDGG, Sasmita GMA, Pratama IPAE. Penetration Testing Berbasis OWASP Testing Guide Versi 4.2 (Studi Kasus: X Website). JITTER J Ilm Teknol dan Komput. 2023;4(1):1613.
5. Anelia SS, Jayanta, Hananto B. Uji Penetrasi Server Universitas PQR Menggunakan Metode National Institute Of Standards And Technology (NIST SP 800-115). J Ilmu Tek dan Komput. 2023;7(1):35–43.
6. Syafaat A. Identifikasi Kerentanan Keamanan Pada Website Fakultas Ilmu Komputer Universitas Subang Menggunakan Metodologi Owasp. E-Journal [Internet]. 2024;11(1):84–99. Available from: http://ejournal.unsub.ac.id/index.php/Fasilkom
7. Shanley A, Johnstone MN. Selection of penetration testing methodologies: A comparison and evaluation. Aust Inf Secur Manag Conf AISM 2015. 2015;2015:65–72.
8. Hidayatulloh S, Saptadiaji D. Penetration Testing pada Website Universitas ARS Menggunakan Open Web Application Security Project (OWASP). J Algoritm. 2021;18(1):77–86.
9. Zahra NA, Zidane FH, Kuslaila NR. Analisis Keamanan Sistem Informasi Pada Website Pt Sentra Vidya Utama (Sevima) Menggunakan Metode Owasp. Pros Semin Nas Teknol dan

Sist Inf. 2023;3(1):384–93.
10. V. Drake. OWASP Web Security Testing Guide v4.2 released [Internet]. 2020. Available from: https://medium.com/@victoriadotdev/owasp-web- security-testing-guide-v4-2-released-7910ea1d7e47
11. Riandhanu IO. Analisis Metode Open Web Application Security Project (OWASP) Menggunakan Penetration Testing pada Keamanan Website Absensi. J Inf dan Teknol. 2022;4(3):160–5.
12. Priambodo DF, Rifansyah AD, Hasbi M. Penetration Testing Web XYZ Berdasarkan OWASP Risk Rating. Teknika. 2023;12(1):33–46.
13. Abdan MK. Pengujian Keamanan Sistem Informasi Berbasis Web Berdasarkan Framework Owasp Wstg V4.2 (Studi Kasus: Sistem Sekawan V1 Universitas Islam Indonesia). Univ Islam Indones [Internet]. 2022;2:1–95. Available from: https://dspace.uii.ac.id/handle/123456789/40200
14. Rafeli AI, Seta HB, Widi IW. Pengujian Celah Keamanan Menggunakan Metode OWASP Web Security Testing Guide (WSTG) pada Website XYZ. Inform J Ilmu Komput. 2022;18(2):97.
15. Kuncoro AW, Rahma F. Analisis Metode Open Web Application Security Project (OWASP) pada Pengujian Keamanan Website: Literature Review. Automata [Internet]. 2021;3(1):1–5. Available from: https://www.sciencedirect.com