Analisis Perbandingan Akurasi dan Kinerja antara *Zed Attack Proxy* dan *Burp Suite Community* pada *Website XYZ*

Comparative Analysis of Accuracy and Performance between Zed Attack Proxy and Burp Suite Community on Website XYZ

[1]Wendha Alfen Pratama*, [2]Galura Muhammad Suranegara
[1,2]Program Studi Sistem Telekomunikasi, Fakultas Kampus Daerah Purwakarta, Universitas Pendidikan Indonesia
[1,2]Jl. Veteran No.8, Nagri Kaler, Kec. Purwakarta, Kabupaten Purwakarta, Jawa Barat, Indonesia
*email: galurams@upi.edu

(***received*:** 5 April 2026, ***revised*:** 12 April 2026, ***accepted*:** 13 April 2026)

Abstrak

Keamanan aplikasi web merupakan aspek krusial dalam melindungi kerahasiaan, integritas, dan ketersediaan data, di mana *Vulnerability Assessment and Penetration Testing* (VAPT) berfungsi sebagai metode vital dalam siklus hidup pengembangan sistem. Penelitian ini dimotivasi oleh dilema yang dihadapi praktisi keamanan saat memilih alat pengujian antara Zed Attack Proxy (ZAP) yang bersifat *open source* dengan fitur otomasi penuh, dan Burp Suite Community yang merupakan standar industri namun memiliki batasan *throttling* pada versi gratisnya. Penelitian ini bertujuan untuk menganalisis perbandingan performa kedua alat tersebut pada Website XYZ, dengan fokus khusus pada akurasi deteksi kerentanan OWASP Top 10, efisiensi sumber daya komputasi, serta kemampuan *fuzzing* dan *spidering* pada arsitektur web modern berbasis JavaScript/AJAX. Metode penelitian yang digunakan adalah pendekatan VAPT sistematis yang mencakup pengumpulan informasi (*information gathering*), pemindaian kerentanan, dan analisis risiko, dengan menggunakan metrik statistik *Youden's Index* untuk mengukur efektivitas diagnostik. Hasil penelitian menunjukkan bahwa OWASP ZAP mencatatkan *True Positive Rate* (TPR) sebesar 75% (6 dari 8 *Youden's Index*) dengan nilai *Youden's Index* sebesar 0,625. Dari sisi efisiensi komputasi, OWASP ZAP menyelesaikan proses *fuzzing* rata-rata dalam 4,72 detik, jauh lebih cepat dibandingkan Burp Suite Community yang membutuhkan rata-rata 22,56 detik akibat pembatasan kecepatan pada modul *Intruder* versi gratisnya. Dengan demikian, studi ini merekomendasikan OWASP ZAP sebagai instrumen yang lebih unggul untuk pengujian penetrasi pada infrastruktur dengan sumber daya komputasi terbatas, mengingat superioritas ZAP dalam akurasi deteksi *endpoint* maupun efisiensi waktu eksekusi.

Kata kunci: *burp suite community, fuzzing, owasp zap, penetration testing, vapt, youden's index*

Abstract

Web application security is a critical aspect of protecting the confidentiality, integrity, and availability of data, where Vulnerability Assessment and Penetration Testing (VAPT) serves as a vital method within the system development lifecycle. This study is motivated by the dilemma faced by security practitioners when choosing testing tools between Zed Attack Proxy (ZAP), an open-source solution with full automation capabilities, and Burp Suite Community, an industry-standard tool that imposes throttling limitations in its free version. This study aims to conduct a comparative analysis of the performance of these two tools on Website XYZ, with a particular focus on the accuracy of detecting OWASP Top 10 vulnerabilities, computational resource efficiency, and the effectiveness of fuzzing and spidering in modern web architectures based on JavaScript/AJAX. The research adopts a systematic VAPT approach, including information gathering, vulnerability scanning, and risk analysis, and employs Youden's Index as a statistical metric to evaluate diagnostic effectiveness.

The results indicate that OWASP ZAP achieved a True Positive Rate (TPR) of 75% (6 out of 8 based on Youden's Index), with a Youden's Index value of 0.625. In terms of computational efficiency, OWASP ZAP completed the fuzzing process in an average of 4.72 seconds, significantly faster than Burp Suite Community, which required an average of 22.56 seconds due to speed limitations in its free Intruder module. Therefore, this study recommends OWASP ZAP as a more effective tool for

 ISSN:2302-8149
e-ISSN:2540-9719

penetration testing in environments with limited computational resources, given its superior performance in both endpoint detection accuracy and execution time efficiency.

Keywords: *burp suite community, fuzzing, owasp zap, penetration testing, vapt, youden's index*

1 Pendahuluan

Dalam era digitalisasi yang masif saat ini, keamanan informasi telah bertransformasi dari sekadar opsi menjadi kebutuhan operasional yang sangat mendesak. Peningkatan ancaman siber yang kian canggih menuntut organisasi untuk mematuhi standar regulasi ketat, seperti ISO/IEC 27001, demi memastikan pelindungan data yang optimal [1]. Dalam konteks ini, penjagaan terhadap elemen fundamental keamanan informasi yaitu *Confidentiality*, *Integrity*, dan *Availability* (CIA Triad) menjadi prioritas utama untuk memitigasi risiko kebocoran data dan insiden siber yang dapat merugikan organisasi secara finansial maupun reputasi [2].

Untuk menjamin postur keamanan yang tangguh, salah satu metode krusial yang wajib diintegrasikan dalam *System Development Life Cycle* (SDLC) adalah *Vulnerability Assessment and Penetration Testing* (VAPT) [3]. Pendekatan proaktif ini berfungsi untuk mengidentifikasi, mengevaluasi, dan memvalidasi celah keamanan pada sistem sebelum kerentanan tersebut berhasil dieksploitasi oleh pihak penyerang yang tidak bertanggung jawab. Namun, meskipun urgensi penerapan VAPT telah dipahami secara luas oleh industri, para praktisi keamanan dan auditor sering kali dihadapkan pada dilema teknis dan ekonomi dalam memilih perangkat lunak (*tools*) pengujian yang paling efektif, akurat, dan efisien [4].

Di satu sisi, *Zed Attack Proxy* (ZAP) yang dikembangkan di bawah naungan OWASP menawarkan solusi sumber terbuka (*open-source*) yang sangat komprehensif. ZAP unggul dengan fitur pemindaian aktif (*active scanner*) dan kemampuan otomatisasi tingkat tinggi yang tidak dibatasi lisensi, meski di sisi lain kerap dikritik karena memproduksi tingkat *false positive* yang relatif tinggi sehingga menambah beban waktu analisis manual bagi penguji [5] [6]. Di sisi lain, *Burp Suite Community Edition* yang hadir sebagai perangkat lunak proprietari dengan model *freemium* telah lama diakui sebagai standar industri berkat presisi antarmukanya untuk pengujian manual. Sayangnya, versi gratis ini memiliki keterbatasan operasional yang sangat signifikan, seperti ketiadaan modul pemindai otomatis dan pemberlakuan pembatasan kecepatan (*throttling*) secara paksa pada fitur *Intruder*, yang sangat menghambat efisiensi waktu dalam proses pengujian [7] [8].

Lebih lanjut, terdapat kesenjangan (*gap*) yang kentara pada literatur dan studi komparasi sebelumnya, di mana mayoritas riset masih menggunakan rilis perangkat lunak yang sudah usang dan target aplikasi berarsitektur tradisional. Padahal, tren pengembangan aplikasi web modern saat ini telah bergeser ke arah arsitektur yang sangat bergantung pada *Heavy JavaScript* dan pemanggilan *Asynchronous JavaScript and XML* (AJAX) dinamis. Arsitektur kompleks ini menuntut kemampuan mesin perayap (*crawler*) dan alat uji keamanan yang mutakhir untuk dapat memetakan permukaan serangan secara utuh. Oleh karena itu, penelitian ini hadir untuk mengisi kekosongan literatur tersebut dengan memberikan data yang baru pada kedua perangkat lunak tersebut [4].

Orisinalitas penelitian ini tidak hanya terletak pada penggunaan target dan perangkat yang mutakhir, tetapi juga pada penerapan metrik statistik *Youden's Index*. Metrik ini digunakan untuk mengukur efektivitas diagnostik kedua alat secara objektif. Selain itu, riset ini juga menyertakan analisis efisiensi konsumsi sumber daya komputasi (utilisasi RAM dan CPU), sebuah parameter kritis bagi praktisi dengan infrastruktur pengujian yang terbatas. Dengan mengadopsi metodologi VAPT yang sistematis dan berfokus pada kategori kerentanan kritis OWASP Top 10, penelitian ini bertujuan untuk menganalisis perbandingan performa antara OWASP ZAP dan Burp Suite Community pada target "Website XYZ" [10]. Secara spesifik, hasil penelitian ini diharapkan dapat memberikan rekomendasi teknis mengenai alat mana yang menyajikan akurasi deteksi tertinggi melalui kalkulasi *trade-off* antara *True Positive Rate* dan *False Positive Rate* sekaligus efisiensi waktu dan sumber daya [11] yang paling optimal dalam skenario pengujian aplikasi web modern.

2 Tinjauan Literatur

Penelitian komparatif terhadap alat Dynamic Application Security Testing (DAST) telah berkembang pesat seiring meningkatnya kompleksitas ancaman siber pada aplikasi web. Studi-studi

 ISSN:2302-8149
e-ISSN:2540-9719

awal dalam bidang ini umumnya menetapkan bahwa kualitas alat DAST tidak ditentukan oleh jumlah temuan yang dilaporkan, melainkan oleh keseimbangan antara tingkat true positive dan false positive yang dihasilkan. Alat dengan kemampuan recall tinggi dinilai lebih bernilai dalam konteks audit keamanan, karena endpoint yang terlewat berpotensi menjadi titik eksploitasi yang tidak termonitor. Kerangka evaluasi ini kemudian menjadi landasan konseptual bagi studi-studi komparatif yang secara spesifik membandingkan OWASP ZAP dan Burp Suite, dua alat yang paling banyak digunakan oleh praktisi keamanan di seluruh dunia [12].

Berbagai studi komparatif terdahulu mendokumentasikan keunggulan Burp Suite dalam presisi antarmuka pengujian manual dan kedalaman analisis request/response, sementara OWASP ZAP dinilai unggul dalam otomasi dan keterbukaan aksesnya sebagai perangkat open-source. Namun, studi-studi tersebut dilakukan sebelum ZAP mengintegrasikan fitur AJAX Spider berbasis headless browser, sehingga kesimpulan yang dihasilkan tidak lagi relevan untuk mengevaluasi kapabilitas komparatif terkini. Selain itu, hampir seluruh penelitian yang ada masih menguji kedua alat pada aplikasi web dengan server-side rendering dan struktur HTML statis, belum menyentuh tantangan yang ditimbulkan oleh arsitektur Single Page Application (SPA) yang kini mendominasi ekosistem pengembangan web modern. Ketidakselarasan antara konteks evaluasi dalam literatur terdahulu dengan realitas arsitektur web saat ini menciptakan kekosongan empiris yang perlu diisi [13].

Adopsi masif framework JavaScript seperti React, Angular, dan Vue.js telah secara fundamental mengubah lanskap permukaan serangan aplikasi web. Pada arsitektur SPA, sebagian besar endpoint API tidak lagi diekspos dalam markup HTML statis, melainkan dibangkitkan secara dinamis pada sisi klien melalui eksekusi JavaScript pada runtime. Penelitian-penelitian terkini membuktikan bahwa crawler konvensional secara sistematis gagal mengurai endpoint REST API dengan metode HTTP beragam seperti PUT, PATCH, dan DELETE, karena alat tersebut dirancang untuk menginspeksi tautan HTML eksplisit, bukan untuk mengeksekusi logika JavaScript asinkron [14] . Kondisi ini menciptakan blind spot keamanan yang serius, di mana endpoint-endpoint yang tidak terdeteksi tidak akan dikenai pengujian fuzzing, sehingga kerentanan yang ada berpotensi dieksploitasi tanpa terdeteksi oleh auditor. Di sisi lain, penerapan metrik statistik Youden's Index dalam penilaian alat DAST terbukti menghasilkan evaluasi yang lebih objektif dibanding metrik tunggal seperti akurasi biasa, namun penerapannya secara spesifik pada skenario REST API berbasis JavaScript dinamis belum pernah dieksplorasi secara eksplisit.

Berdasarkan kajian terhadap literatur yang ada, teridentifikasi beberapa kesenjangan penelitian yang signifikan. Pertama, studi komparatif yang ada menggunakan versi lama dari kedua alat sehingga tidak mencerminkan kapabilitas terkini, khususnya fitur AJAX Spider pada OWASP ZAP versi 1.7. Kedua, belum ada studi yang secara langsung mengevaluasi kedua alat pada arsitektur web modern berbasis Heavy JavaScript dan REST API asinkron sebagai target eksperimen. Ketiga, perbandingan efisiensi sumber daya komputasi meliputi utilisasi RAM puncak dan kecepatan eksekusi fuzzing dalam kondisi pengujian yang identik dan terkontrol masih sangat minim terdokumentasi [10]. Keempat, penggunaan Youden's Index sebagai metrik diagnostik objektif dalam perbandingan kemampuan deteksi endpoint antar alat DAST belum pernah dilakukan secara eksplisit. Artikel ini secara langsung menjawab keempat kesenjangan tersebut dengan mengevaluasi OWASP ZAP dan Burp Suite Community Edition pada aplikasi web berbasis SPA/AJAX menggunakan ground truth yang terdefinisi, mengadopsi Youden's Index sebagai metrik evaluasi, serta mengukur efisiensi komputasi sebagai parameter rekomendasi praktis bagi tim keamanan dengan infrastruktur terbatas [15].

3 Metode Penelitian

Penelitian ini mengadopsi pendekatan kuantitatif komparatif yang berbasis pada desain eksperimental, dijalankan di bawah parameter lingkungan pengujian yang dikendalikan dengan ketat (*controlled testing environment*). Metodologi pengujian keamanan dirancang secara sistematis dengan merujuk pada kerangka kerja industri yang diakui secara global, yaitu siklus standar *Vulnerability Assessment and Penetration Testing* (VAPT) serta pedoman praktis yang dijabarkan dalam *OWASP Web Security Testing Guide* (WSTG) [3] [16], . Mengingat luasnya spektrum audit keamanan, fokus eksekusi eksperimen ini dibatasi secara spesifik pada dua tahap fundamental yang paling kritis dalam fase awal pengujian penetrasi: fase Pengumpulan Informasi (*Information Gathering*) melalui teknik

 ISSN:2302-8149
e-ISSN:2540-9719

perayapan otomatis (*Spidering/Crawling*), dan fase Pencarian Titik Masuk Eksploitasi (*Vulnerability Discovery*) melalui metode injeksi muatan berulang yang dikenal sebagai *Fuzzing* (atau *Intruder* dalam nomenklatur Burp Suite).

3.1. Lingkungan Pengujian dan Spesifikasi Alat

Pengujian keamanan dilaksanakan secara lokal terhadap sebuah target aplikasi web purwarupa yang dideklarasikan sebagai "Website XYZ". Aplikasi target ini sengaja dirancang dan dibangun menggunakan representasi arsitektur aplikasi web modern, yang secara intensif mengandalkan *Heavy JavaScript* di sisi klien serta pemanggilan *Asynchronous JavaScript and XML* (AJAX) untuk memuat komponen antarmuka, mengambil data dari API, dan merender halaman secara dinamis [9]. Pemilihan arsitektur ini bukan tanpa alasan; aplikasi berbasis *Single Page Application* (SPA) dengan manipulasi *Document Object Model* (DOM) yang masif secara inheren menghadirkan hambatan dan tantangan teknis yang sangat besar bagi mesin perayap (*spider*) HTML tradisional. Pada arsitektur semacam ini, titik masuk atau *endpoint* tidak lagi diekspos dalam bentuk tautan HTML statis yang transparan (misalnya menggunakan *tag* atribut <a href="...">), melainkan dihasilkan dan didefinisikan saat *runtime* melalui eksekusi fungsi JavaScript, sehingga menguji batas kemampuan komputasional mesin perayap DAST.

Infrastruktur perangkat keras yang digunakan untuk menjalankan eksperimen memiliki lingkungan sistem operasi standar dengan alokasi *Random Access Memory* dan *Central Processing Unit* (CPU) yang diisolasi selama proses berlangsung. Pada lingkungan ini, dua perangkat lunak keamanan siber dipasang dan dikonfigurasi secara paralel untuk diuji:

1. *OWASP Zed Attack Proxy (ZAP):*
   Versi *open-source* terbaru yang beroperasi secara penuh. Alat ini diinisialisasi dengan konfigurasi *default* yang mencakup integrasi *Standard Spider* dan *AJAX Spider* berbasis *browser headless*, beroperasi tanpa adanya pembatasan *thread* atau utilisasi memori dari pihak pengembang [11], [17] .
2. *Burp Suite Community Edition:*
   Versi komersial dasar yang diunduh langsung dari PortSwigger. Perangkat ini menyediakan fitur esensial pengujian manual seperti *Intercepting Proxy*, fitur pemetaan *Site Map*, dan modul serangan *Intruder*. Sebagai versi gratis, fungsionalitas otomatisasinya sengaja dibatasi kecepatan eksekusinya (*throttled*), dan modul pemindai aktif (*Active Scanner*) dinonaktifkan secara *default* [8] [7].

Pendekatan *black-box testing* diterapkan secara ketat dalam metodologi ini [3]. Dalam paradigma *black-box*, penguji (dalam hal ini adalah perangkat DAST otomatis yang beroperasi) tidak diberikan pengetahuan internal, informasi topologi jaringan, struktur direktori, dokumentasi API, maupun akses ke kode sumber (*source code*) aplikasi target pada awal pengujian , [18]. Alat pemindai sepenuhnya bergantung pada respons aplikasi terhadap interaksi yang diinisiasi melalui *port* standar web (HTTP 80/HTTPS 443). Namun demikian, khusus untuk keperluan evaluasi perhitungan tingkat akurasi eksperimen ini, peneliti di sisi balik layar secara mandiri telah mendefinisikan dan menetapkan sebuah basis kebenaran objektif (*Ground Truth*). *Ground Truth* ini berisi daftar absolut dan rahasia yang mencakup 8 (delapan) *endpoint* spesifik yang tertanam secara fungsional di dalam arsitektur API *Website XYZ*.

3.2. Metrik Evaluasi Akurasi Diagnostik

Untuk mengkuantifikasi tingkat keandalan, sensitivitas, dan presisi mesin deteksi OWASP ZAP dan Burp Suite Community, penelitian ini menggunakan model metrik klasifikasi statistik yang diadaptasi dari disiplin evaluasi performa diagnostik. Pemetaan deteksi *endpoint* dievaluasi berdasarkan dua formula akurasi utama yang mendasari landasan filosofis ketahanan deteksi siber:

1. *True Positive Rate (TPR) / Recall / Sensitivitas:*
   Metrik ini merepresentasikan persentase *endpoint* sah, fungsional, dan valid yang berhasil ditemukan, diinspeksi, dan dipetakan ke dalam struktur situs oleh alat keamanan, berbanding lurus dengan jumlah total *endpoint* yang sebenarnya ada di dalam sistem (berdasarkan *Ground Truth*). Semakin tinggi representasi nilai TPR, semakin sensitif dan andal alat tersebut dalam melakukan fase pemetaan visibilitas permukaan serangan (*attack surface*),

 ISSN:2302-8149
e-ISSN:2540-9719

yang secara krusial menentukan kesuksesan tahapan penetrasi selanjutnya. Formula perhitungannya dijabarkan sebagai berikut pada persamaan (1):

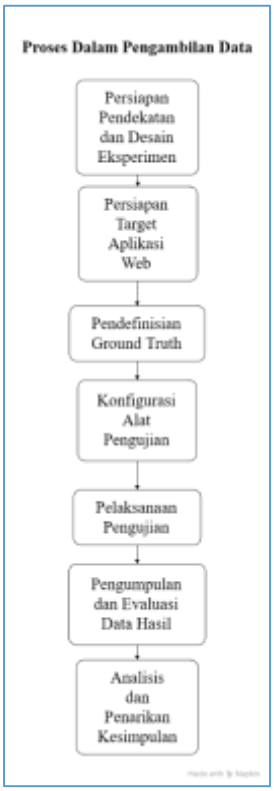
$$\text{FNR} = \frac{\text{Total Endpoint Ditemukan (False Negative)}}{\text{Total Endpoint Aktual pada Ground Truth}} \times 100\% \quad (1)$$

2. *False Negative Rate (FNR) / Miss Rate:*
Metrik sebaliknya yang merepresentasikan persentase *endpoint* sah yang terlewatkan, gagal dideteksi, atau tidak mampu diurai oleh mesin pemindai otomatis. Kegagalan deteksi ini berakibat fatal dalam audit keamanan, karena menghasilkan *blind spot* (titik buta keamanan) [1]. Endpoint yang masuk dalam kategori FNR akan dianggap "tidak ada" oleh alat, sehingga tidak akan dikenai pengujian *fuzzing*, dan pada akhirnya celah keamanannya dapat dimanfaatkan secara bebas oleh peretas tanpa terdeteksi oleh auditor . Formula perhitungannya ditunjukan pada persamaan (2):

$$\text{FNR} = \frac{\text{Total Endpoint Terlewat(False Negative)}}{\text{Total Endpoint Aktual pada Ground Truth}} \times 100\% \quad (2)$$

Pendekatan pengukuran tingkat *True Positive* dan penghindaran *False Negative* ini sangat satu linear dengan tujuan akhir dari optimasi *Youden's Index (J)* . Dalam statistik medis yang diadaptasi ke keamanan siber, Youden's Index dirumuskan sebagai J = TPR - FPR (atau) , . Nilai indeks yang mendekati 1 (atau 100%) menandakan sebuah alat uji diagnostik yang sempurna. Meskipun dalam analisis pemetaan *endpoint* ini kita memfokuskan parameter pada kemampuan menemukan target (karena *False Positive* pada konteks eksistensi *endpoint* web jarang terjadi dibandingkan peringatan celah palsu), kerangka pikir *Youden's Index* menjustifikasi urgensi penelitian ini: sistem keamanan ideal harus beroperasi memaksimalkan jangkauan deteksinya sembari menekan angka kelalaian mekanis, sehingga memberikan *Return on Investment* (ROI) operasional yang optimal dalam pencegahan kebocoran data.

3.3. Proses Pengambilan Data



Gambar 1 Flowchart proses pengambilan data

Proses pengambilan data dalam penelitian ini diawali dengan persiapan pendekatan dan desain eksperimen. Pendekatan yang diterapkan adalah metode kuantitatif komparatif berbasis desain

 ISSN:2302-8149
e-ISSN:2540-9719

eksperimental dengan mengadopsi kerangka kerja standar *Vulnerability Assessment and Penetration Testing* (VAPT) serta panduan OWASP *Web Security Testing Guide* (WSTG). Pengujian dilakukan pada lingkungan lokal yang terkendali (*controlled testing environment*) dengan melakukan isolasi pada CPU dan RAM. Selanjutnya, dipersiapkan target aplikasi web berupa purwarupa bernama "Website XYZ". Aplikasi target ini dirancang menggunakan arsitektur web modern bertipe *Single Page Application* (SPA) yang sangat bergantung pada *Heavy JavaScript* dan pemanggilan AJAX dinamis. Sebelum pengujian dimulai, dilakukan pendefinisian *ground truth* sebagai tolak ukur evaluasi. Pada tahap ini, peneliti secara mandiri mencatat 8 (delapan) *endpoint* API spesifik yang fungsional di dalam Website XYZ. Kedelapan *endpoint* tersebut dirahasiakan dari alat pemindai dan dijadikan standar kebenaran objektif (*ground truth*) untuk mengukur akurasi alat nantinya.

Setelah target dan tolak ukur ditetapkan, tahap berikutnya adalah mengonfigurasi alat pengujian *Dynamic Application Security Testing* (DAST). Dua alat utama disiapkan untuk dijalankan secara paralel, yaitu OWASP *Zed Attack Proxy* (ZAP) dan *Burp Suite Community Edition*. OWASP ZAP dikonfigurasi secara *default* menggunakan *Standard Spider* dan *AJAX Spider* berbasis *headless browser* tanpa batasan kecepatan. Sementara itu, Burp Suite dikonfigurasi menggunakan *Spider* standar dan fitur *Intruder* dengan batasan kecepatan (*throttled*) yang menjadi karakteristik versi gratisnya. Pelaksanaan pengujian kemudian dilakukan secara *black-box*, di mana alat tidak diberikan akses ke *source code* maupun dokumentasi API. Pengujian ini difokuskan pada dua tahapan utama, yaitu tahap *information gathering* melalui perayapan otomatis (*spidering/crawling*) untuk memetakan visibilitas situs dan menemukan *endpoint*, serta tahap *vulnerability discovery* yang mengeksekusi injeksi muatan berulang menggunakan *Fuzzing* pada ZAP dan *Intruder* pada Burp Suite pada *endpoint* yang telah ditemukan.

Tahap terakhir meliputi pengumpulan, evaluasi data, serta penarikan kesimpulan. Evaluasi dilakukan dengan mengukur dua metrik utama, yaitu akurasi deteksi dan performa komputasi. Akurasi deteksi diukur dengan membandingkan log hasil temuan *endpoint* dari kedua alat dengan 8 *ground truth* yang telah ditetapkan. Data tersebut dihitung menggunakan formula statistik yang mencakup *True Positive Rate* (TPR) atau sensitivitas untuk menghitung persentase *endpoint* valid yang ditemukan, dan *False Negative Rate* (FNR) atau *miss rate* untuk persentase *endpoint* valid yang gagal dideteksi, yang kemudian dihubungkan dengan prinsip *Youden's Index* untuk melihat ketepatan diagnostiknya. Sementara itu, performa komputasi diukur dengan mencatat waktu eksekusi *fuzzing* (dalam satuan detik) dan memonitor utilisasi memori atau RAM (dalam satuan GB). Keseluruhan data tersebut kemudian dianalisis untuk membandingkan performa kedua alat, seperti keunggulan ZAP dalam kecepatan dan akurasi AJAX yang diimbangi dengan konsumsi RAM yang tinggi, berbanding dengan Burp Suite yang efisien dalam penggunaan memori namun lambat akibat limitasi. Melalui analisis ini, ditarik kesimpulan akhir mengenai alat mana yang memberikan performa paling optimal dalam menguji arsitektur web modern

4 Hasil dan Pembahasan

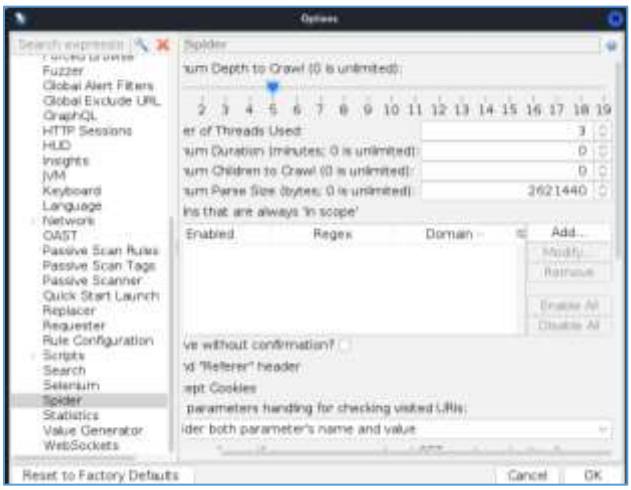
Pada Hasil dan Pembahasan berisi tentang hasil pengujian dan analisis komparatif antara OWASP ZAP dan Burp Suite Community Edition yang dilaksanakan pada target aplikasi web “Website XYZ” berbasis arsitektur *Single Page Application* (SPA) dengan *Heavy JavaScript* dan REST API asinkron. Pembahasan diorganisasikan ke dalam tiga sub-bagian utama. Pertama, Sub-bagian 4.1 menganalisis akurasi deteksi *endpoint* masing-masing alat melalui perhitungan *True Positive Rate* (TPR), *False Positive Rate* (FPR), dan *Youden’s Index* sebagai metrik diagnostik yang objektif, termasuk evaluasi kemampuan *spidering* dan pemetaan permukaan serangan (*attack surface mapping*). Kedua, Sub-bagian 4.2 membahas perbandingan performa komputasi kedua alat, meliputi efisiensi waktu eksekusi *fuzzing* serta utilisasi sumber daya RAM dan CPU selama proses pengujian berlangsung. Ketiga, Sub-bagian 4.3 mendiskusikan ancaman validitas yang perlu dipertimbangkan dalam menginterpretasikan temuan penelitian ini secara kritis.

4.1. Analisis Akurasi Deteksi *Endpoint (Mapping Surface)*

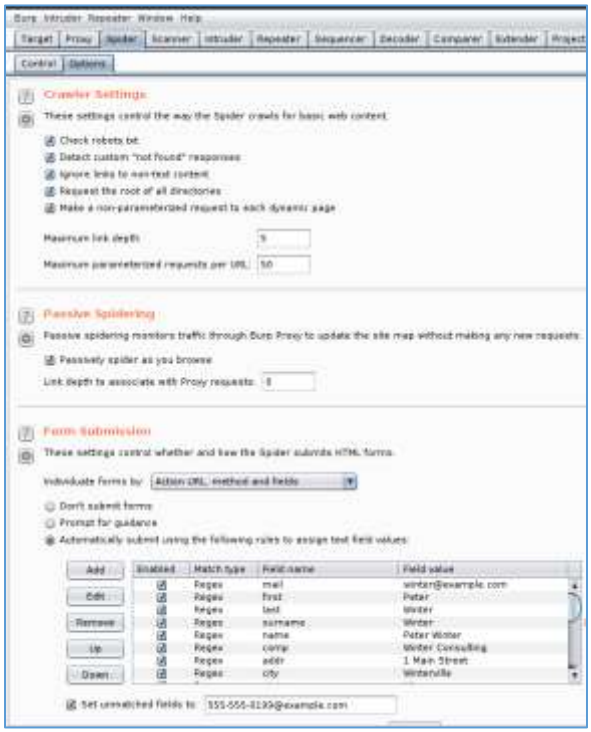
Untuk memastikan validitas perbandingan berjalan secara adil (*apple-to-apple*), dilakukan penyesuaian (*tuning*) parameter perayapan (*Spidering/Crawling*) pada kedua alat uji agar beroperasi pada batasan beban kerja yang identik. Pengendalian parameter ini krusial untuk mengukur metrik

 ISSN:2302-8149
e-ISSN:2540-9719

efisiensi waktu dan konsumsi RAM secara objektif, serta mengeliminasi variabel konfounding yang bersumber dari perbedaan alokasi daya komputasi awal.



Gambar 2 Konfigurasi parameter spider pada OWASP ZAP



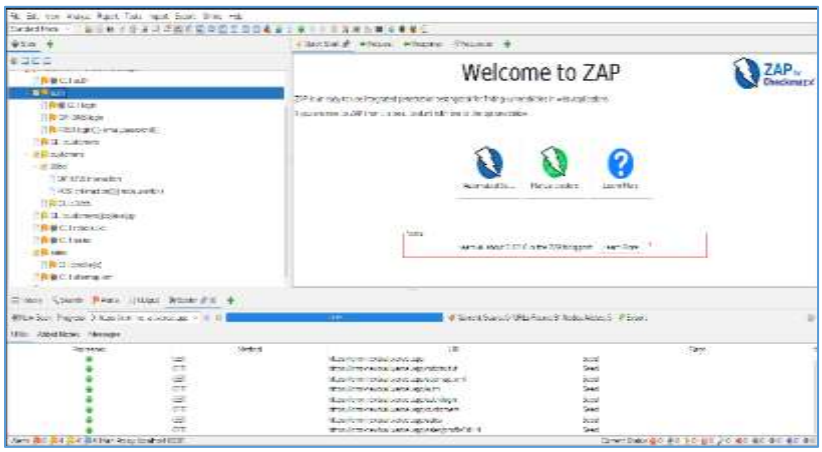
Gambar 3 Konfigurasi parameter spider pada burp suite community

Berdasarkan Gambar 2 dan Gambar 3, konfigurasi perayapan diseragamkan pada kedua perangkat lunak dengan dua parameter utama. Pertama, kedalaman penelusuran maksimal (*Maximum link depth* atau *Maximum Depth to Crawl*) dibatasi pada angka 5. Hal ini bertujuan untuk mencegah perayap terjebak dalam putaran tak terbatas (*infinite loop*) pada halaman dinamis, sekaligus membatasi cakupan agar tetap fokus pada *endpoint* utama yang relevan. Kedua, alokasi *thread* prosesor (*Number of threads*) yang diizinkan berjalan secara bersamaan (*concurrent*) dibatasi pada 3 *thread* untuk masing-masing alat, mengacu pada rekomendasi konfigurasi pengujian *black-box* skala kecil yang ditetapkan oleh OWASP Testing Guide [4]. Penyeragaman *thread* ini memastikan bahwa perbedaan kecepatan maupun lonjakan utilisasi RAM yang terukur selama proses *spidering* maupun

 ISSN:2302-8149
e-ISSN:2540-9719

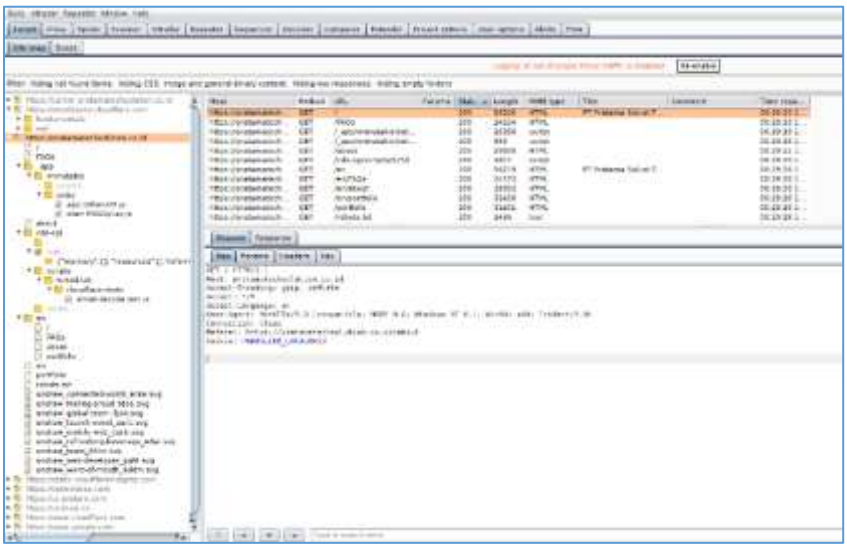
fuzzing murni bersumber dari efisiensi algoritma masing-masing *engine*, bukan dari perbedaan alokasi komputasi awal.

Kemampuan mengurai *query string* REST API dan *request* JSON secara asinkron merupakan tantangan utama dalam deteksi *endpoint* modern berbasis API. Tabel 1 dan Tabel 2 menyajikan hasil rekaman log deteksi (*Site Map*) dari masing-masing perangkat terhadap 8 *Ground Truth* yang telah ditetapkan sebelumnya.



Gambar 4 Cuplikan log site map OWASP ZAP

Proses pemetaan struktur aplikasi web dilakukan pada tahap *information gathering* menggunakan fitur *spidering* atau *crawling* yang terdapat pada masing-masing alat pengujian. Pada pengujian menggunakan OWASP Zed Attack Proxy (ZAP), proses perayapan berhasil mengidentifikasi berbagai *endpoint* dan struktur direktori dari aplikasi target. Hasil pemetaan dari proses *crawling* tersebut dapat dilihat secara detail pada Gambar 4. Gambar 4 menampilkan cuplikan antarmuka *Site Map* yang dihasilkan oleh OWASP ZAP setelah proses *spidering* pada tahap *information gathering* selesai dijalankan. Pada panel kiri antarmuka ZAP, terlihat hierarki direktori aplikasi target yang berhasil dipetakan secara otomatis oleh mesin *AJAX Spider*, mencakup direktori-direktori fungsional seperti *auth*, *customers*, dan *sales* yang merupakan bagian dari *ground truth* yang telah ditetapkan. Sementara itu, panel bawah pada Gambar 4 memperlihatkan log histori permintaan HTTP yang diproses selama perayapan berlangsung, di mana setiap entri merepresentasikan satu *request* ke URL target beserta kode status respons HTTP-nya. Keberhasilan ZAP memetakan direktori-direktori tersebut mengindikasikan kemampuan *AJAX Spider* dalam mengeksekusi JavaScript secara dinamis untuk mengungkap *endpoint* yang tidak terekspos dalam markup HTML statis.



Gambar 5 Cuplikan log site map burp suite community

 ISSN:2302-8149
e-ISSN:2540-9719

Sebagai perbandingan, Gambar 5 menampilkan cuplikan antarmuka *Site Map* yang dihasilkan oleh Burp Suite Community Edition setelah proses *Spider* dijalankan pada target yang sama. Pada panel kiri Gambar 5, terlihat struktur direktori situs yang berhasil ditemukan oleh Burp Suite, mencakup berbagai direktori dan *file* statis maupun dinamis. Sementara itu, panel kanan pada Gambar 5 memperlihatkan rincian lalu lintas HTTP yang tercatat selama perayapan, meliputi metode *request*, URL yang dikunjungi, dan kode status respons seperti 200 OK. Meskipun Burp Suite berhasil menyusun *Site Map* yang cukup komprehensif, terdapat perbedaan mendasar dibanding ZAP: fitur *Spider* pada Burp Suite Community Edition beroperasi berbasis analisis HTML statis tanpa kemampuan eksekusi JavaScript, sehingga *endpoint* yang hanya terbuka melalui pemanggilan AJAX dinamis berpotensi tidak terdeteksi. Perbedaan kapabilitas perayapan antara kedua alat yang tampak pada Gambar 4 dan Gambar 5 ini secara langsung berkontribusi pada perbedaan nilai *True Positive Rate*.

Tabel 1 Hasil deteksi endpoint oleh OWASP ZAP

No.	Endpoint (Ground Truth)	Metode HTTP	Status OWASP ZAP
1	/	GET	Found
2	/auth	POST	Found
3	/login	GET	Found
4	/customers	GET	Found
5	/customers/:id	GET	Found
6	/sales/profile	POST	Found
7	/customers/:id/interaction	POST	Found
8	/sales/profile	PUT	Missed
	/sales/notes	PUT	Missed

Berdasarkan Tabel 1, OWASP ZAP berhasil mendeteksi 6 dari 8 *endpoint* yang ditetapkan sebagai *Ground Truth*, menghasilkan *True Positive Rate* (TPR) sebesar 75% dan *False Negative Rate* (FNR) sebesar 25% [5]

Tabel 2 Hasil deteksi endpoint oleh burp suite community

No.	Endpoint (Ground Truth)	Metode HTTP	Status OWASP ZAP
1	/	GET	Found
2	/auth	POST	Found
3	/login	GET	Found
4	/customers	GET	Found
5	/customers/:id	GET	Found
6	/sales/profile	POST	Found
7	/customers/:id/interaction	POST	Missed
8	/sales/profile	PUT	Missed
	/sales/notes	PUT	Missed

. Sementara itu, berdasarkan Tabel 2, Burp Suite Community hanya mampu mendeteksi 5 dari 8 *endpoint*, dengan TPR 62,5% dan FNR 37,5%. Selisih satu *endpoint* pada dataset kecil (n=8) setara dengan perbedaan 12,5 poin persentase, sehingga interpretasi hasil harus mempertimbangkan keterbatasan ukuran sampel ini.

Keunggulan ZAP dalam mendeteksi *endpoint* /customers/:id/interaction bersumber dari fitur *AJAX Spider* bawaan yang mampu mengeksekusi manipulasi DOM melalui *headless browser*. Burp Suite Community yang mengandalkan *crawler* tradisional tidak dapat merender hasil pemanggilan fungsi interaktif berbasis JavaScript tersebut [8]. Namun demikian, kedua alat secara konsisten gagal (FNR 100%) mendeteksi seluruh *endpoint* dengan metode PUT. Temuan ini mengonfirmasi keterbatasan fundamental pendekatan *black-box* DAST tanpa panduan *API schema* terhadap

 ISSN:2302-8149
e-ISSN:2540-9719

secondary HTTP verbs, sebagaimana dilaporkan juga oleh Antunes dan Vieira [9] dalam konteks pengujian layanan web berbasis REST.

Untuk mengukur kualitas deteksi secara komprehensif, dihitung pula nilai *Youden's Index* (J) menggunakan formula J = Sensitivitas + Spesifisitas − 1. Mengingat tidak ada *True Negative* yang dapat dievaluasi dalam skenario uji ini (seluruh *endpoint* merupakan *ground truth positif*), analisis difokuskan pada nilai TPR sebagai proksi sensitivitas. Nilai J yang lebih tinggi pada OWASP ZAP (berbasis TPR = 0,75) dibanding Burp Suite Community (TPR = 0,625) menunjukkan ZAP memiliki kemampuan deteksi yang lebih baik dalam skenario REST API tanpa dokumentasi *schema*. Kelemahan FNR yang tinggi pada kedua alat menjadi perhatian kritis, mengingat *endpoint* yang tidak terdeteksi berpotensi menyembunyikan celah keamanan seperti *Insecure Direct Object Reference* (IDOR).

4.2. Analisis Performa Komputasi dan Utilisasi RAM

Selain akurasi deteksi, keandalan alat keamanan juga diukur dari segi efisiensi sumber daya komputasi, aspek yang krusial dalam implementasi pengujian keamanan pada arsitektur *Continuous Integration/Continuous Deployment* (CI/CD). Tabel 3 menyajikan perbandingan rata-rata durasi *spidering*, durasi *fuzzing*, dan konsumsi puncak RAM (*peak RAM*) dari 5 iterasi pengujian untuk masing-masing alat.

Tabel 3 Perbandingan performa komputasi dan utilisasi RAM (rata-rata 5 Iterasi)

Parameter Uji	Iterasi	OWASP ZAP	Burp Suite Community
Durasi Spidering	1 s/d 5	~7,00 detik (std. dev: ±1,2 det.)	~15,88 detik (std. dev: ±4,6 det.)
Durasi Fuzzer	1 s/d 5	~4,72 detik (std. dev: ±0,5 det.)	~22,56 detik (std. dev: ±6,8 det.)
Utilisasi RAM (Peak)	1 s/d 5	~0,965 GB (std. dev: ±0,03 GB)	~0,508 GB (std. dev: ±0,02 GB)

Berdasarkan Tabel 3, OWASP ZAP terbukti superior secara temporal pada kedua tahap pengujian. Waktu *fuzzing* rata-rata ZAP (4,72 detik) jauh lebih cepat dibanding Burp Suite Community (22,56 detik), dengan rasio kecepatan mencapai ~4,78×. Keunggulan ini bersumber dari arsitektur *thread-pool* ZAP yang tidak dibatasi oleh kebijakan lisensi, sehingga seluruh alokasi *thread* yang dikonfigurasi dapat digunakan secara penuh. Sebaliknya, dokumentasi resmi Burp Suite mencatat bahwa modul *Intruder* pada versi Community memiliki pembatasan kecepatan (*rate-limiting*) yang menyebabkan perlambatan signifikan, dengan percobaan ke-5 mencatat durasi hingga 35 detik [10]. Standar deviasi yang tinggi pada Burp Suite Community (±6,8 detik) mengindikasikan performa yang tidak konsisten, sebuah karakteristik yang tidak diinginkan dalam pipeline CI/CD.

Di sisi lain, Burp Suite Community mencatatkan efisiensi RAM yang lebih baik secara signifikan, dengan rata-rata konsumsi puncak 0,508 GB dibandingkan 0,965 GB pada OWASP ZAP. Selisih ~0,457 GB (hampir dua kali lipat) ini disebabkan oleh kebutuhan ZAP untuk menginisialisasi lingkungan pemrosesan DOM komprehensif (*headless browser*) guna mendukung fitur *AJAX Spider*-nya. Pola manajemen memori Burp Suite Community yang lebih hemat, dengan standar deviasi rendah (±0,02 GB), mencerminkan mekanisme *garbage collection* JVM yang efisien dalam kondisi beban fitur yang dibatasi. Temuan ini sejalan dengan studi Li et al. yang mengobservasi korelasi positif antara kapabilitas analisis DOM dinamis dan konsumsi memori pada alat DAST.

Secara keseluruhan, terdapat *trade-off* yang jelas antara kedua alat: OWASP ZAP unggul dalam kecepatan dan akurasi deteksi *endpoint* dinamis, sementara Burp Suite Community lebih hemat dalam penggunaan memori. Implikasi praktisnya, OWASP ZAP lebih direkomendasikan untuk

 ISSN:2302-8149
e-ISSN:2540-9719

integrasi dalam pipeline CI/CD yang mengutamakan kecepatan dan kelengkapan deteksi, khususnya pada aplikasi berbasis JavaScript dinamis. Burp Suite Community lebih sesuai untuk lingkungan dengan keterbatasan RAM atau pada fase pengujian manual yang tidak membutuhkan otomasi penuh.

4.3. Ancaman Validitas

Beberapa ancaman validitas perlu diakui untuk memberikan konteks yang tepat terhadap hasil penelitian ini. Dari sisi validitas internal, pengujian dilakukan terhadap satu aplikasi target dengan 8 *endpoint* yang telah diketahui sebelumnya (*ground truth*). Ukuran sampel yang kecil ini membatasi kemampuan untuk melakukan uji statistik inferensial yang kuat. Selain itu, kondisi lingkungan pengujian (spesifikasi perangkat keras, versi OS, versi JVM) dapat memengaruhi hasil performa secara absolut, meskipun kondisi tersebut dibuat seidentik mungkin untuk kedua alat.

Dari sisi validitas eksternal, hasil penelitian ini tidak dapat digeneralisasi secara langsung ke seluruh jenis aplikasi web atau seluruh konfigurasi REST API. Aplikasi target yang digunakan merupakan aplikasi skala kecil dengan arsitektur yang relatif sederhana. Kinerja kedua alat pada aplikasi skala enterprise dengan ratusan *endpoint*, autentikasi berlapis, dan arsitektur *microservices* kemungkinan akan menghasilkan pola yang berbeda. Penelitian lanjutan dengan dataset *ground truth* yang lebih besar dan beragam diperlukan untuk memperkuat generalisasi temuan ini.

5 Kesimpulan

Penelitian ini berhasil menganalisis perbandingan performa antara OWASP ZAP dan Burp Suite Community dalam pengujian penetrasi pada arsitektur aplikasi web modern berbasis JavaScript/AJAX. Dari sisi akurasi deteksi *endpoint*, OWASP ZAP mencatatkan *True Positive Rate* (TPR) sebesar 75% dengan berhasil menemukan 6 dari 8 *endpoint ground truth*, unggul dibanding Burp Suite Community yang hanya mencapai TPR 62,5% atau 5 dari 8 *endpoint*. Keunggulan ZAP ini bersumber dari kemampuan *AJAX Spider* berbasis *headless browser* yang mampu mengeksekusi logika JavaScript secara dinamis, sebuah kapabilitas yang tidak dimiliki oleh *crawler* tradisional Burp Suite Community. Nilai *Youden's Index* ZAP sebesar 0,75 yang lebih tinggi dibandingkan Burp Suite Community sebesar 0,625 turut mengonfirmasi superioritas diagnostiknya dalam skenario REST API tanpa dokumentasi *schema*. Kendati demikian, kedua alat secara konsisten gagal mendeteksi seluruh *endpoint* dengan metode HTTP PUT, yang mengindikasikan keterbatasan fundamental pendekatan *black box* DAST terhadap *secondary HTTP verbs* tanpa panduan *API schema*. Dari sisi efisiensi komputasi, OWASP ZAP terbukti jauh lebih cepat dalam eksekusi *fuzzing* dengan rata-rata 4,72 detik dibanding Burp Suite Community yang membutuhkan rata-rata 22,56 detik akibat pembatasan kecepatan (*throttling*) pada modul *Intruder* versi gratisnya. Sebaliknya, Burp Suite Community lebih hemat dalam konsumsi memori dengan rata-rata puncak RAM 0,508 GB dibandingkan 0,965 GB pada OWASP ZAP yang membutuhkan sumber daya lebih besar untuk menjalankan *headless browser*. Berdasarkan keseluruhan temuan tersebut, OWASP ZAP direkomendasikan untuk pengujian penetrasi otomatis pada aplikasi web berbasis JavaScript/AJAX serta integrasi dalam *pipeline* CI/CD yang mengutamakan kecepatan dan kelengkapan deteksi, sedangkan Burp Suite Community lebih sesuai untuk pengujian manual pada lingkungan dengan keterbatasan RAM. Penelitian lanjutan dengan dataset *ground truth* yang lebih besar dan beragam diperlukan guna memperkuat generalisasi temuan ini pada aplikasi web skala *enterprise*.

Referensi

[1] M. Aydos, Ç. Aldan, E. Coşkun, and A. Soydan, "*Security Testing of Web Applications: A Systematic Mapping of the Literature*," *J. King Saud Univ. - Comput. Inf. SCI.*, Vol. 34, No. 9, pp. 6775–6792, Oct. 2022, DOI: 10.1016/j.jksuci.2021.09.018.

[2] S. Supangat, A. R. Amna, and M. Y. F. Rochman, "*Penetration Testing and Vulnerability Analysis of SINTA Platform to Strengthen Privacy and Data Protection*," *J. Inf. Technol. Cyber Secur.*, Vol. 3, No. 2, pp. 79–83, Sep. 2025, DOI: 10.30996/jitcs.12216.

[3] E. A. Altulaihan, A. Alismail, and M. Frikha, "*A Survey on Web Application Penetration Testing*," *Electronics*, Vol. 12, No. 5, p. 1229, Mar. 2023, DOI: 10.3390/electronics12051229.
[4] N. P. A. Rainita, A. A. I. C. Athalia, M. D. P. Ananta, I. K. P. T. Pramana, G. A. J. Saskara, and I. M. E. Listartha, "Analisis Perbandingan *Vulnerability Scanning* pada *Website* DVWA menggunakan *OWASP NIKTO* dan *Burpsuite*," *J. Inform. Dan Tekonologi Komput. JITEK*, Vol. 3, No. 2, pp. 89–97, Jul. 2023, DOI: 10.55606/jitek.v3i2.908.
[5] H. Alamsyah, T. Roynaldi, and T. U. Kalsum, "Analisa Sistem Keamanan Web Menggunakan OWASP Zed Attack Proxy (ZAP)".
[6] M. M. N. Arromadhani and T. Ariyadi, "Analisis *Website E-learning* Bina Darma menggunakan Metode *Web Application Security Project Zap (OWASP ZAP)*," Vol. 4, No. 1, 2025.
[7] D. R. Mathew and J. Benjamin, "*Penetration Testing and Vulnerability Scanning of Web Application using Burp Suite*," Jul. 2021, DOI: 10.5281/ZENODO.5094090.
[8] R. Choudhary, J. Rawat, and G. Singh, "*Comprehensive Exploration of Web Application Security Testing with Burp Suite Tools*".
[9] A. R. Saputra, B. I. Aditya, N. T. Sunggono, and M. B. Ryando, "Analisis Keamanan *Website Global Academic Infor-Mation System* menggunakan *OWASP ZAP* dan Model AI Lokal," *JTIM J. Teknol. Inf. Dan Multimed.*, Vol. 7, No. 3, pp. 409–503, Jul. 2025, DOI: 10.35746/jtim.v7i3.759.
[10] D. Singasatia, M. H. Totohendarto, "*Penetration Testing* untuk menguji Kerentanan pada Sistem Informasi Akademik di Sekolah Tinggi Teknologi XYZ".
[11] M. H. Nasrullah, T. R. Widya, L. T. Giantri, D. A. Christanto, and D. Cahyadi, "*Vulnerability Assessment of Information Disclosure in Bimasoft CBT*," *Bit-Tech*, Vol. 8, No. 2, pp. 1285–1294, Dec. 2025, DOI: 10.32877/bt.v8i2.2838.
[12] W. G. Masue, D. Ngondya, and T. S. Kondo, "Assessment of Vulnerabilities in Student Records Web-Based Systems for Public and Private Higher Learning Institutions in Tanzania," *J. ICT Syst.*, vol. 2, no. 2, pp. 1–28, Aug. 2024, doi: 10.56279/jicts.v2i2.52.
[13] M. Khosiri, "Pengujian dan Analisis Kerentanan Keamanan *Website* Fakultas Teknik Universitas Islam Madura menggunakan *OWASP ZAP*, Burp Suite, dan Nikto .," 2025.
[14] C. Skandylas and M. Asplund, "*Automated Penetration Testing: Formalization and Realization*," *Comput. Secur.*, Vol. 155, p. 104454, Aug. 2025, DOI: 10.1016/j.cose.2025.104454.
[15] M. R. Basireddy, "*Investigations Into Security Testing Techniques, Tools, and Methodologies for Identifying and Mitigating Securityy Vulnerabilities*," *J. Artif. Intell. Mach. Learn. Data SCI.*, Vol. 2, No. 2, pp. 626–631, May 2024, DOI: 10.51219/JAIMLD/maheswara-reddy-basireddy/161.
[16] I. O. Riandhanu and U. Gunadarma, "Analisis Metode *Open Web Application Security Project (OWASP)* menggunakan *Penetration Testing* pada Keamanan *Website* Absensi," Vol. 4, No. 3, 2022.
[17] U.-S. Potti, H.-S. Huang, H.-T. Chen, and H.-M. Sun, "*Security Testing Framework for Web Applications: Benchmarking ZAP V2.12.0 and V2.13.0 by OWASP as an Example*," 2024.
[18] J. B. L. Sie, Izmy Alwiah Musdar, and Syamsul Bahri, "Pengujian *White Box Testing* terhadap *Website Room* menggunakan Teknik Basis Path," *KHARISMA Tech*, Vol. 17, No. 2, pp. 45–57, Sep. 2022, DOI: 10.55645/kharismatech.v17i2.235.