

# Implementasi Wazuh Untuk Deteksi Anomali dan Notifikasi Real-Time Menggunakan Whatsapp Pada Server E-Commerce

A. A. Pareza<sup>1</sup>, L. Widyawati<sup>2</sup>, K. Abd. Latif<sup>3</sup>

<sup>1,2,3</sup> Program Studi Teknologi Informasi, Fakultas Teknik, Universitas Bumigora  
e-mail: [ejafareza07@gmail.com](mailto:ejafareza07@gmail.com)

## Abstrak

Perkembangan e-commerce yang pesat meningkatkan risiko serangan siber pada server, seperti Denial of Service (DoS) dan Brute Force, sehingga dibutuhkan sistem deteksi anomali dan respons insiden yang cepat dan real-time. Penelitian ini bertujuan mengimplementasikan Wazuh sebagai Security Information and Event Management (SIEM) yang diintegrasikan dengan WhatsApp sebagai media notifikasi dan mitigasi real-time pada server e-commerce. Metode penelitian yang digunakan adalah studi kasus dan Secure Software Development Life Cycle (SSDLC), dengan tahapan mulai dari analisis kebutuhan, perancangan sistem, pengembangan, hingga pengujian menggunakan White-Box Testing, simulasi serangan DoS, dan Brute Force. Sistem dibangun dengan mengintegrasikan Wazuh SIEM, API Gateway berbasis Node.js (whatsapp-web.js), dan mekanisme active response untuk pemblokiran IP penyerang melalui firewall. Hasil penelitian menunjukkan bahwa sistem berhasil mendeteksi serangan DoS dan Brute Force secara akurat, mengirimkan notifikasi real-time ke administrator melalui WhatsApp, serta menyediakan mekanisme mitigasi interaktif dua arah (human-in-the-loop) berupa balasan pesan untuk memblokir atau mengabaikan IP penyerang. Pengujian White-Box pada fungsi-fungsi kritis sistem juga menunjukkan bahwa seluruh jalur logika dan penanganan kesalahan berjalan sesuai ekspektasi, dengan tingkat keberhasilan 19 dari 19 basis path (100%). Dengan demikian, integrasi Wazuh SIEM dan WhatsApp sebagai gateway notifikasi dan mitigasi terbukti efektif meningkatkan kecepatan deteksi dan respons terhadap insiden keamanan pada server e-commerce.

**Kata Kunci:** SIEM, Wazuh, Whatsapp, E-Commerce, Keamanan Siber

## Abstract

The rapid growth of e-commerce increased the risk of cyber attacks on servers, such as Denial of Service (DoS) and Brute Force, so that a fast and real-time anomaly detection and incident response system was required. This study aimed to implement Wazuh as a Security Information and Event Management (SIEM) platform integrated with WhatsApp as a real-time notification and mitigation medium for an e-commerce server. The research combined a case study method with the Secure Software Development Life Cycle (SSDLC), covering requirement analysis, system design, development, and testing using White-Box Testing along with DoS and Brute Force attack simulations. The system was built by integrating the Wazuh SIEM, a Node.js-based API Gateway (whatsapp-web.js), and an active response mechanism to block attacker IP addresses through the firewall. The results showed that the system successfully detected DoS and Brute Force attacks accurately, sent real-time notifications to the administrator via WhatsApp, and provided an interactive two-way, human-in-the-loop mitigation mechanism through reply messages to block or ignore the attacker IP. White-Box testing on the critical system functions also showed that all logic paths and error handling behaved as expected, achieving a 100% success rate across 19 basis paths. Overall, the integration of Wazuh SIEM and WhatsApp as a notification and mitigation gateway is proven effective in improving the speed of detection and response to security incidents on e-commerce servers.

**Keywords:** SIEM, Wazuh, Whatsapp, E-Commerce, Cyber Security

## LATAR BELAKANG

Perkembangan e-commerce yang pesat memberikan kemudahan transaksi online bagi masyarakat, namun di sisi lain meningkatkan risiko serangan siber karena platform ini menyimpan data sensitif seperti identitas pelanggan, transaksi keuangan, dan kredensial login (Kamil & Miranda, 2024). Kerentanan ini terbukti pada kasus peretasan e-commerce Coupang, di mana peretas berhasil mengeksploitasi celah keamanan server internal hingga menyebabkan kebocoran data pribadi milik 33,7 juta pengguna (CNBC Indonesia, 2026). Maraknya ancaman siber seperti malware, brute force, dan eksploitasi celah keamanan menegaskan bahwa deteksi dini terhadap aktivitas mencurigakan merupakan aspek yang sangat krusial bagi keberlangsungan sistem (Fachri, 2023).

Salah satu ancaman yang sering dihadapi oleh server adalah Denial of Service (DoS), yaitu serangan yang bertujuan membuat layanan server tidak dapat diakses dengan membanjiri server menggunakan lalu lintas palsu dalam jumlah besar (Nedyalkov, 2024). Pada tahun 2024, sistem pertahanan DoS otonom Cloudflare memblokir sekitar 21,3 juta serangan DoS, meningkat 53% dibandingkan tahun 2023, dengan rata-rata 4.870 serangan diblokir setiap jamnya. Skala serangan yang begitu besar membuat kebutuhan akan sistem monitoring keamanan yang tangguh dan real-time menjadi semakin mendesak (Syaiikhurrahman & Prasetyo, 2025).

Salah satu solusi yang dapat digunakan adalah Security Information and Event Management (SIEM), yang berfungsi mengumpulkan, menganalisis, dan mengelola log dari berbagai sumber dalam infrastruktur TI (Coppolino et al., 2022). Salah satu platform SIEM open-source yang populer adalah Wazuh, yang menawarkan fitur log analysis, file integrity monitoring, intrusion detection, dan alerting untuk membantu administrator mengidentifikasi dan merespons potensi ancaman dengan cepat (Saputra et al., 2024).

Secara arsitektural, Wazuh terdiri atas tiga komponen utama yang saling terintegrasi, yaitu Wazuh Agent yang dipasang pada perangkat target untuk melakukan monitoring lokal dan mengirimkan log secara terenkripsi, Wazuh Server (Manager) yang bertindak sebagai otak pengolah data dengan mengeksekusi mesin korelasi berbasis ruleset, serta Wazuh Indexer dan Dashboard yang berfungsi sebagai mesin pencarian sekaligus antarmuka visualisasi bagi analis keamanan (Padmavathi & Saminathan, 2025). Selain kemampuan analisis log dan file integrity monitoring, Wazuh juga mendukung fitur active response yang memungkinkan eksekusi skrip mitigasi secara otomatis, misalnya memblokir alamat IP penyerang pada firewall lokal (Kouari et al., 2025). Kapabilitas inilah yang menjadi dasar bagi penelitian ini untuk mengembangkan mekanisme notifikasi dan mitigasi yang terhubung langsung dengan platform WhatsApp.

Penelitian terkait implementasi Wazuh telah dilakukan dalam beberapa konteks, namun masih menyisakan celah signifikan. (Kurnia et al., 2025) memanfaatkan Wazuh File Integrity Monitoring untuk memantau perubahan tidak sah pada file, tetapi belum mengintegrasikan notifikasi real-time dan belum diujikan pada lingkungan e-commerce. mengembangkan Wazuh FIM di lingkungan akademik dengan visualisasi dashboard, namun tidak menyediakan mekanisme notifikasi otomatis maupun evaluasi pada sistem yang memerlukan deteksi anomali cepat. (Shafiyyah et al., 2024) telah mengintegrasikan Wazuh dan WhatsApp serta fail2ban untuk mitigasi brute force dan DoS, namun cakupannya terbatas pada ancaman jaringan tanpa mempertimbangkan kebutuhan spesifik e-commerce seperti proteksi payment gateway. Dengan demikian, gap penelitian terletak pada integrasi Wazuh dengan notifikasi real-time dan respons otomatis yang terpersonalisasi untuk e-commerce (Tafonao & Raslina, 2025).

Penelitian ini hadir untuk mengisi celah tersebut dengan menggabungkan kemampuan deteksi anomali Wazuh secara menyeluruh dengan integrasi notifikasi real-time melalui WhatsApp yang dilengkapi opsi tindakan (actions), serta menerapkannya pada lingkungan server e-commerce yang memiliki kebutuhan keamanan lebih ketat dan dinamis (F. P. E. Putra et al., 2025). Berdasarkan latar belakang tersebut, rumusan masalah penelitian ini adalah

bagaimana cara melakukan monitoring, deteksi anomali, dan notifikasi real-time pada server e-commerce dengan memanfaatkan Wazuh dan WhatsApp. Penelitian ini bertujuan untuk mengimplementasikan Wazuh sebagai sistem deteksi anomali dan notifikasi real-time yang terintegrasi dengan WhatsApp pada server e-commerce, menganalisis efektivitasnya dalam mendeteksi ancaman keamanan, serta mengevaluasi kecepatan dan keakuratan sistem dalam memberikan peringatan kepada administrator dan melakukan mitigasi secara interaktif (D. T. Putra et al., n.d.).

## METODE

Penelitian ini menggunakan dua pendekatan metodologis yang saling melengkapi, yaitu metode studi kasus untuk mengumpulkan data dan mengimplementasikan langsung sistem yang dibangun pada web e-commerce dummy, serta metode Secure Software Development Life Cycle (SSDLC) yang telah disesuaikan dengan kebutuhan penelitian. Studi kasus digunakan untuk mendalami fenomena keamanan siber secara kontekstual pada infrastruktur aplikasi web e-commerce yang berjalan di atas server Linux, sedangkan SSDLC digunakan sebagai kerangka kerja pengembangan sistem yang menitikberatkan keamanan sejak tahap awal perancangan (*security by design*) (Zahid et al., 2023).

Objek studi kasus adalah aplikasi web e-commerce dummy bernama NextMerce yang dibangun menggunakan framework Next.js dan dirancang menyerupai sistem e-commerce nyata, meliputi fitur login, katalog produk, dan checkout. Penggunaan web dummy dipilih karena pertimbangan etis dan legal, yaitu untuk menghindari pengujian langsung pada situs e-commerce produksi. Lingkungan studi kasus dibangun di atas jaringan virtual terisolasi yang terdiri dari empat komponen utama: (1) Target Server yang menjalankan aplikasi web e-commerce, Node.js API Gateway (SOAR), Wazuh Agent, dan firewall lokal; (2) Wazuh Manager sebagai SIEM Server yang menerima dan menganalisis log secara real-time; (3) Attacker Node berupa mesin Kali Linux yang menjalankan perkakas Siege untuk simulasi DoS dan skrip brute force untuk simulasi serangan pada endpoint login; serta (4) Administrator Client berupa WhatsApp SOAR Gateway yang menerima notifikasi insiden dan mengeksekusi keputusan mitigasi.

Skenario kerja sistem dimulai ketika penyerang melancarkan simulasi serangan ke server target, sehingga seluruh aktivitas terekam pada berkas log lokal. Wazuh Agent membaca log tersebut secara real-time dan meneruskannya ke Wazuh Manager, yang kemudian mencocokkan pola aktivitas dengan ruleset berbasis XML. Apabila aktivitas dinyatakan sebagai ancaman valid, Wazuh Manager memicu peringatan dan meneruskannya melalui HTTP POST Webhook ke Node.js API Gateway. Gateway kemudian mengekstraksi informasi IP penyerang dan deskripsi ancaman, lalu mengirimkan notifikasi interaktif ke WhatsApp administrator berupa dua opsi balasan, yaitu 1 (Blokir) atau 2 (Abaikan). Ketika administrator membalas dengan opsi 1, sistem mengeksekusi perintah pemblokiran pada firewall lokal (*iptables* pada Linux atau *New-NetFirewallRule* pada Windows) sehingga seluruh paket data dari IP penyerang ditolak dan server kembali pada kondisi aman.

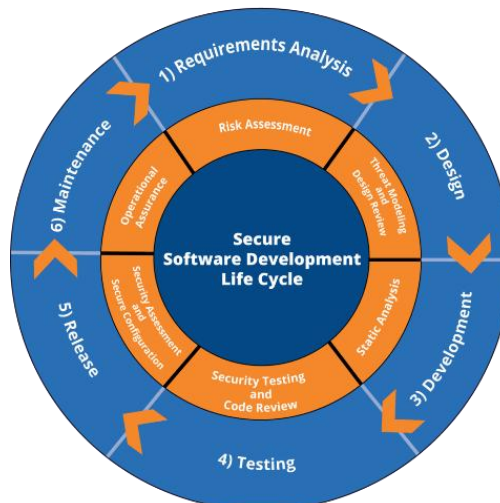
Pengumpulan data pada penelitian ini dilakukan melalui dua teknik utama. Pertama, studi literatur (data sekunder) yang mencakup kajian jurnal penelitian terdahulu, dokumentasi teknis Wazuh, Node.js, dan Docker, serta standar keamanan seperti OWASP Top 10, yang digunakan sebagai acuan dalam memetakan potensi risiko pada tahap requirement analysis dan merancang pola serangan simulasi yang relevan dengan kasus aplikasi e-commerce (Alamin & Amirul, 2025). Kedua, observasi dan eksperimen langsung pada lingkungan laboratorium simulasi, yang meliputi pengumpulan berkas log mentah aktivitas web server dan otentikasi sistem, data alert JSON yang dihasilkan oleh Wazuh Manager, data waktu respons (*delay/latency*) pengiriman notifikasi dari API Gateway ke WhatsApp, serta berkas log eksekusi firewall sebagai bukti keberhasilan pemblokiran IP penyerang.

Sebagai bagian dari tahap Requirement Analysis pada SSDLC, dilakukan Risk Assessment untuk mengidentifikasi potensi ancaman berdasarkan tingkat kemungkinan (*likelihood*) dan dampak (*impact*), sebagaimana disajikan pada Tabel 1.

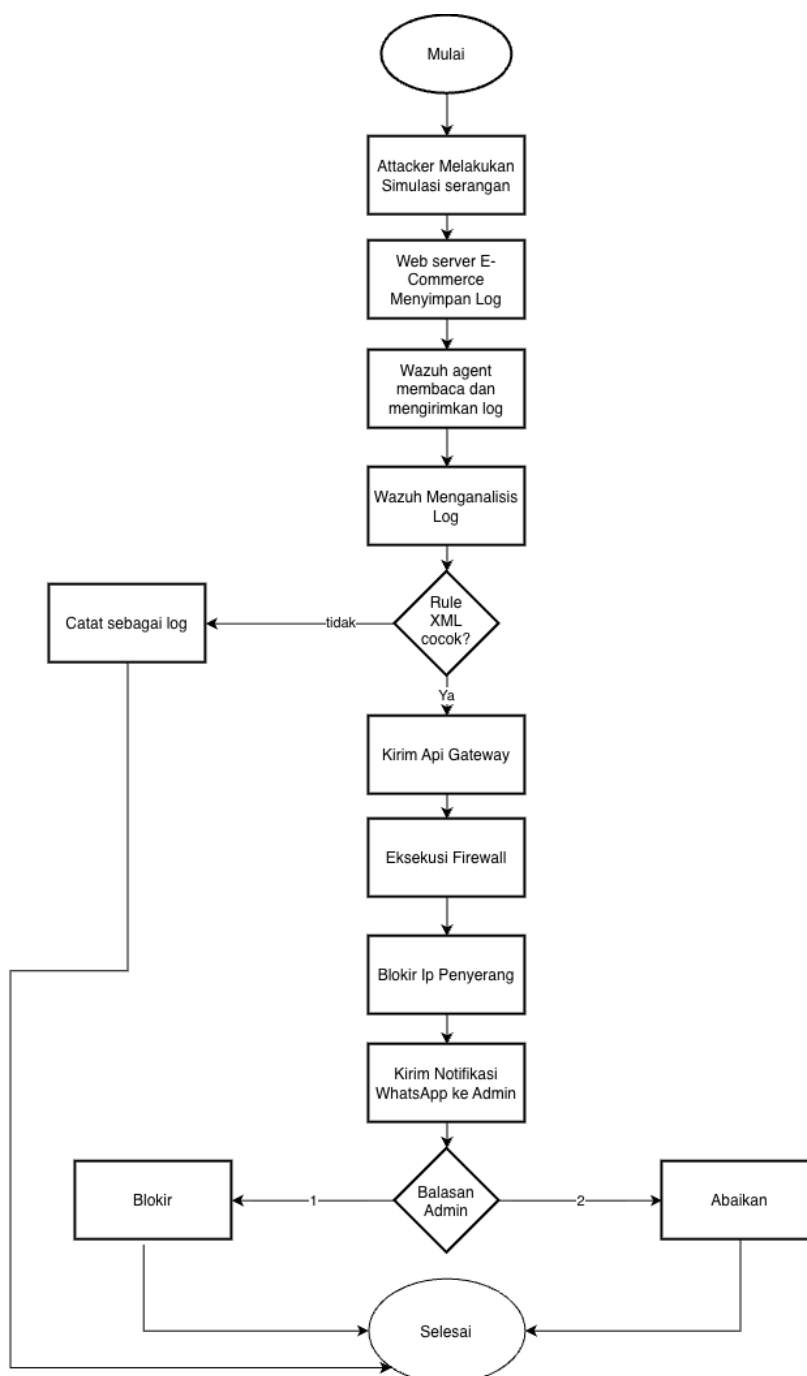
**Tabel 1. Hasil Risk Assessment**

| ID   | Potensi Ancaman / Kerentanan                                            | Likelihood | Impact |
|------|-------------------------------------------------------------------------|------------|--------|
| R-01 | Serangan Brute Force pada layanan autentikasi server                    | Tinggi     | Sedang |
| R-02 | Eksplorasi SQL Injection pada aplikasi web e-commerce                   | Sedang     | Tinggi |
| R-03 | Keterlambatan waktu respons (Incident Response Time) oleh administrator | Tinggi     | Tinggi |

Berdasarkan hasil Risk Assessment pada Tabel 1, kebutuhan utama sistem yang dibangun adalah menghadirkan mekanisme deteksi yang mampu mengenali pola serangan R-01 dan R-02, serta menyediakan kontrol respons aktif untuk meminimalkan dampak risiko R-03. Kebutuhan perangkat lunak yang digunakan meliputi Visual Studio Code, sistem operasi Linux dan Windows, Node.js, bahasa pemrograman JavaScript dan Python, serta Docker, sedangkan kebutuhan perangkat keras berupa laptop dengan prosesor Intel Core i7, RAM 8 GB, dan penyimpanan SSD NVMe 512 GB. SSDLC yang diterapkan pada penelitian ini dibatasi pada empat tahap awal, yaitu requirement analysis, design, development, dan testing, sebagaimana disajikan pada Gambar 1.

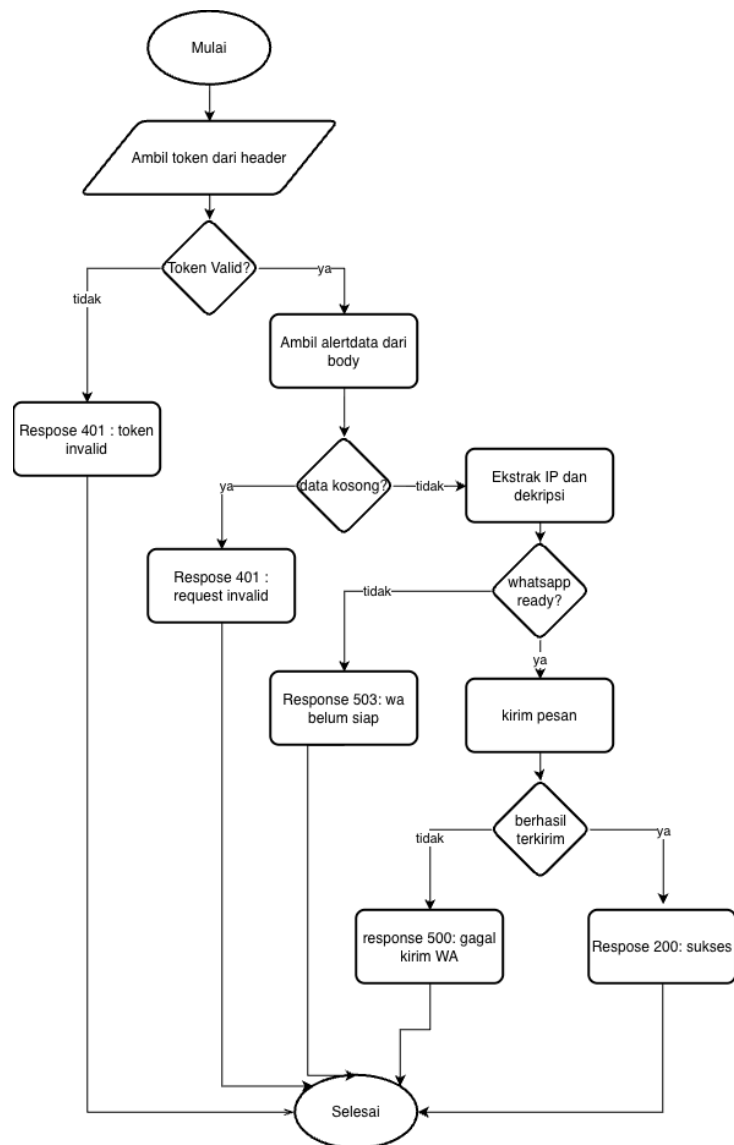
**Gambar 1. Tahapan SSDLC yang Digunakan pada Penelitian**

Pada tahap design, dilakukan threat modeling untuk memetakan alur ancaman dan skenario mitigasi otomatis dan interaktif (Security Orchestration, Automation, and Response / SOAR) yang menghubungkan entitas penyerang, infrastruktur server, sistem SIEM, hingga intervensi manusia (human-in-the-loop), sebagaimana diilustrasikan pada Gambar 2. Perancangan ini mencakup tiga alur logika utama, yaitu alur pemrosesan webhook dan pengiriman notifikasi WhatsApp, alur eksekusi pemblokiran IP pada firewall yang mendeteksi platform sistem operasi secara otomatis (cross-platform detection), serta alur pemrosesan pesan balasan interaktif dari administrator melalui WhatsApp.



**Gambar 2. Diagram Alur Pemodelan Ancaman dan Mitigasi Respons Insiden Berbasis WhatsApp Interaktif**

Secara lebih rinci, Gambar 3 menggambarkan flowchart logika pemrosesan permintaan (incoming request processing) pada API Gateway Webhook. Alur dimulai dari validasi header token x-wazuh-token; apabila token tidak sesuai, sistem langsung mengembalikan status HTTP 401 dan menghentikan proses. Apabila token valid, sistem memeriksa isi req.body; jika kosong, sistem menerapkan mekanisme fallback berupa data dummy simulasi Brute Force, sedangkan jika terisi, data alert asli digunakan. Sistem kemudian mengekstraksi alamat IP penyerang dan deskripsi ancaman untuk disimpan pada variabel memori global, memeriksa status kesiapan klien WhatsApp (whatsappReady), lalu mengirimkan pesan notifikasi terstruktur. Kegagalan pengiriman direspons dengan HTTP 500, sedangkan keberhasilan direspons dengan HTTP 200 beserta detail ID pesan.

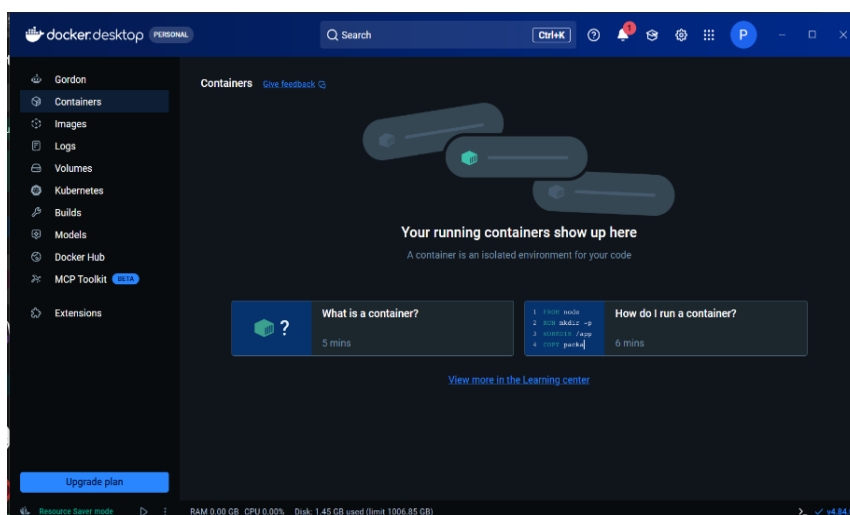


**Gambar 3. Alur Pemrosesan Webhook dan Pengiriman Notifikasi WhatsApp**

Pengujian sistem dilakukan melalui dua pendekatan, yaitu White-Box Testing menggunakan metode Basis Path Testing untuk memverifikasi seluruh jalur logika internal pada fungsi-fungsi kritis (blockIPInFirewall, message handler WhatsApp, dan endpoint webhook), serta pengujian keamanan berbasis skenario nyata berupa simulasi serangan DoS menggunakan perkakas Siege dan simulasi Brute Force menggunakan skrip pengujian kredensial pada endpoint login.

## HASIL DAN PEMBAHASAN

Tahap development diawali dengan penyiapan infrastruktur berbasis containerisasi menggunakan Docker Desktop yang dijalankan di atas Windows Subsystem for Linux 2 (WSL 2) dengan distribusi Ubuntu. Repositori resmi wazuh-docker versi 4.8.0 diunduh melalui Git, kemudian komponen Wazuh Manager dan Filebeat di dalam kontainer diperbarui ke versi 4.14.7 agar kompatibel dengan kebutuhan penelitian. Gambar 4 menunjukkan antarmuka Docker Desktop yang digunakan untuk memantau dan mengelola siklus hidup seluruh kontainer, meliputi Wazuh Manager, Wazuh Indexer, dan Wazuh Dashboard.



**Gambar 4. Tampilan Antarmuka Docker Desktop**

Setelah infrastruktur Wazuh berjalan, dilakukan pengembangan skrip integrasi berbasis Python pada direktori `/var/ossec/integrations/custom-whatsapp` untuk menjembatani Wazuh Manager dengan Node.js API Gateway. Skrip ini menerima tiga parameter dari daemon `wazuh-integratord`, yaitu jalur berkas alert, token autentikasi, dan URL webhook, kemudian mengekstraksi alamat IP penyerang menggunakan mekanisme multi-structure fallback yang memindai atribut `clientip` maupun `srcip` pada payload JSON (Ozkan-okay et al., 2023). Data alert yang telah dinormalisasi selanjutnya dikirimkan ke endpoint webhook Node.js melalui HTTP POST dengan batas waktu lima detik, dan setiap tahapan eksekusi dicatat ke dalam berkas `audit integrations.log`.

Konfigurasi integrasi pada Wazuh Manager didefinisikan untuk mendengarkan Rule ID 100002, 100003, 100020 (indikasi DoS/HTTP Request Flooding) dan 100011 (indikasi Brute Force Login), dengan format alert berupa JSON dan otentikasi menggunakan API key. Sebagai fondasi deteksi, dirancang ruleset kustom pada berkas `local_rules.xml` yang dikelompokkan dalam grup `nextjs,ecommerce,security_event`. Rule 100002 bertindak sebagai base rule yang mendekomposisi log aktivitas aplikasi Next.js. Rule 100010 mendeteksi setiap kegagalan login tunggal, sedangkan Rule 100011 merupakan aturan korelasi tingkat tinggi (level 12) yang terpicu apabila terdapat lima kali kegagalan login dalam 60 detik dari IP sumber yang sama dan dipetakan ke teknik MITRE ATT&CK T1110. Rule 100020 merupakan aturan ambang batas yang terpicu apabila terdapat 100 permintaan dalam 10 detik dari IP yang sama, dipetakan ke teknik T1498, sebagaimana disajikan pada potongan konfigurasi berikut.

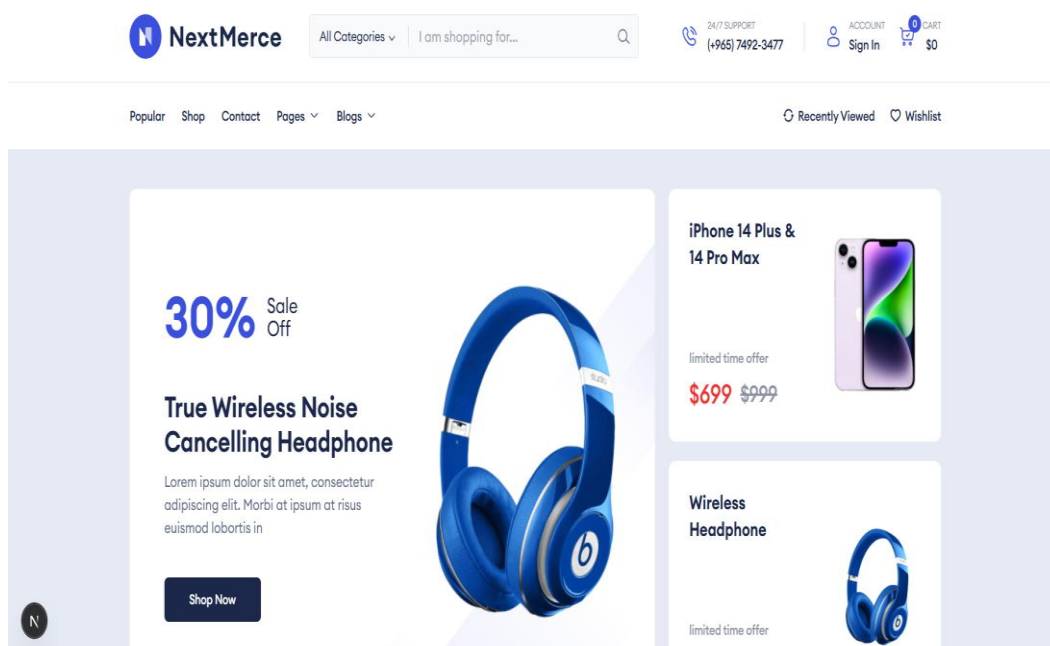
```
<rule id="100011" level="12" frequency="5" timeframe="60">
<if_matched_sid>100010</if_matched_sid> <same_source_ip/> <description>SECURITY
ALERT: Indikasi Brute Force Login dari IP $(srcip)!</description> </rule>
```

Pada sisi API Gateway, dikembangkan endpoint webhook berbasis Express.js yang melakukan validasi token melalui header `x-wazuh-token`, mengekstraksi data alert, memeriksa status kesiapan klien WhatsApp (`whatsappReady`), dan mengirimkan pesan notifikasi terstruktur ke administrator melalui fungsi `sendWhatsAppMessage`. Fungsi `blockIPInFirewall(ip)` diimplementasikan sebagai modul active response berbasis JavaScript Promise yang memvalidasi format IP menggunakan regular expression, mendeteksi platform sistem operasi (Cross-Platform Detection) melalui `os.platform()`, kemudian menyusun dan mengeksekusi perintah firewall yang sesuai, yaitu `iptables` pada Linux atau `New-NetFirewallRule` pada Windows, melalui `child process.exec` dengan batas waktu 30 detik. Balasan interaktif dari administrator ditangani melalui event `waClient.on('message')`, yang memvalidasi nomor pengirim (`isAdminNumber`), kemudian menerjemahkan balasan angka 1 sebagai instruksi blokir dan angka 2 sebagai instruksi mengabaikan alert, di samping menyediakan perintah teks bantuan (`menu/help`) dan status sistem.

```
if (platform === 'linux') { command = `sudo iptables -A INPUT -s ${ip} -j DROP`; } else
if (platform === 'win32') { command = `New-NetFirewallRule -DisplayName
'Block_Attacker_${ip}' -Direction Inbound -Action Block -RemoteAddress ${ip}`; }
```

Potongan program di atas memperlihatkan bagian inti dari fungsi blockIPInFirewall(ip) yang menyusun perintah firewall secara dinamis berdasarkan hasil deteksi platform sistem operasi (Cross-Platform Detection). Pendekatan ini memastikan satu basis kode dapat digunakan pada dua sistem operasi berbeda tanpa duplikasi logika, sekaligus menjaga agar proses pemblokiran tetap non-blocking terhadap layanan utama API Gateway karena dijalankan melalui child\_process.exec secara asinkronus di dalam objek Promise. Pada sisi penanganan pesan balasan, event waClient.on('message') memverifikasi nomor pengirim melalui fungsi isAdminNumber() sebelum mengizinkan eksekusi perintah apa pun, sehingga pesan yang berasal dari nomor selain administrator akan diabaikan tanpa balasan. Desain ini menempatkan verifikasi identitas sebagai lapisan otorisasi pertama sebelum sistem mengizinkan perintah interaktif (1/2/menu/status) memicu tindakan pada firewall, sehingga risiko penyalahgunaan perintah oleh pihak yang tidak berwenang dapat diminimalkan.

Sebagai objek pengujian, disiapkan aplikasi web e-commerce dummy NextMerce (Gambar 5) yang menyediakan katalog produk, pencarian, serta modul otentikasi yang dikonfigurasi untuk mencatat setiap aktivitas transaksi dan trafik masuk ke dalam format log JSON terstruktur. Dua titik krusial dijadikan fokus pengujian, yaitu endpoint otentikasi (halaman signin) sebagai target simulasi Brute Force dan halaman utama (root/landing page) sebagai target simulasi HTTP Flooding untuk memicu Rule ID 100020. Setiap kegagalan login pada halaman signin secara otomatis menghasilkan baris log JSON terstruktur berisi indikator LOGIN\_FAILED beserta atribut IP sumber, yang menjadi prasyarat krusial bagi Wazuh Agent untuk melakukan korelasi ambang batas pada Rule ID 100011.



**Gambar 5. Tampilan Antarmuka Beranda Aplikasi Web E-Commerce NextMerce Pengujian White-Box**

Pengujian white-box dilakukan menggunakan metode Basis Path Testing untuk memastikan seluruh jalur independen pada struktur kendali program telah dieksekusi minimal satu kali. Pada fungsi blockIPInFirewall diperoleh nilai Cyclomatic Complexity  $V(G) = 4$  berdasarkan tiga node keputusan (validasi format IP, deteksi platform, dan evaluasi hasil eksekusi), dengan lima basis path yang seluruhnya menunjukkan hasil sesuai. Pada handler pesan WhatsApp waClient.on('message'), diuji delapan basis path yang mencakup verifikasi nomor administrator, opsi balasan 1 dan 2, perintah bantuan, status sistem, hingga penanganan exception, dengan seluruh jalur berhasil sesuai ekspektasi. Pada endpoint webhook POST /test/webhook, diuji enam basis path yang mencakup validasi token, fallback data dummy, serta status koneksi WhatsApp, dan seluruhnya berjalan sesuai rancangan. Rekapitulasi hasil pengujian white-box disajikan pada Tabel 2.

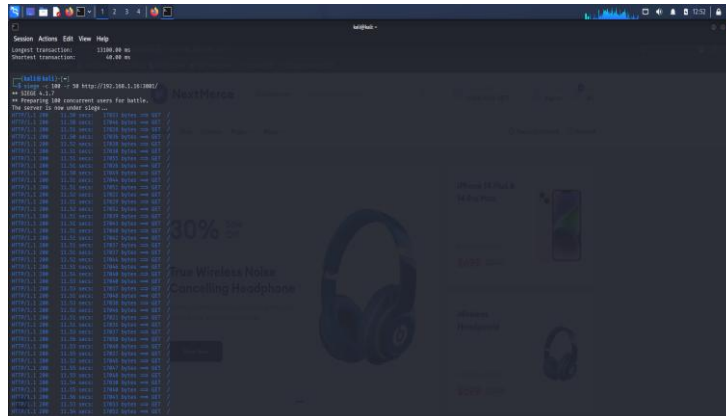
**Tabel 2. Rekapitulasi Hasil Pengujian White-Box**

| Objek Pengujian        | Ruang Lingkup Analisis                                                                 | Tingkat Keberhasilan |
|------------------------|----------------------------------------------------------------------------------------|----------------------|
| blockIPInFirewall      | 5 basis path (validasi format IP, deteksi OS, eksekusi CLI, penanganan sukses/gagal)   | 5/5 sesuai (100%)    |
| waClient.on('message') | 8 basis path (verifikasi admin, menu interaktif 1/2, help, status, exception handling) | 8/8 sesuai (100%)    |
| POST /test/webhook     | 6 basis path (validasi token, fallback payload, status koneksi WA, status pengiriman)  | 6/6 sesuai (100%)    |

Rekapitulasi pada Tabel 2 menunjukkan bahwa keseluruhan 19 basis path dari tiga komponen inti sistem berhasil dieksekusi sesuai ekspektasi (tingkat keberhasilan 100%), tanpa ditemukan penyimpangan alur logika (Sie et al., 2022). Hasil ini menjadi dasar keyakinan bahwa mekanisme deteksi hingga eksekusi mitigasi otomatis dapat diandalkan sebelum dilanjutkan ke tahap pengujian dengan skenario serangan nyata.

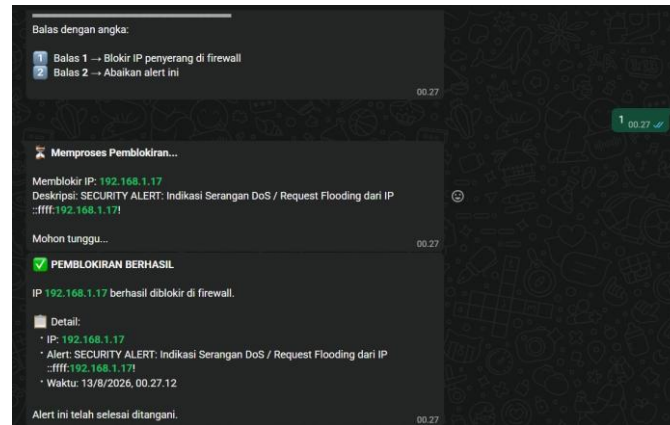
#### **Pengujian Denial of Service (DoS)**

Pengujian DoS dilakukan dari mesin Kali Linux menggunakan perkakas Siege dengan parameter `-c 100 -r 50` yang ditujukan ke endpoint `http://192.168.1.16:3001/`, sehingga menghasilkan akumulasi 5.000 permintaan HTTP GET dalam kurun waktu singkat, sebagaimana ditunjukkan pada Gambar 6. Lonjakan permintaan ini berhasil memenuhi ambang batas Rule ID 100020, yaitu lebih dari 100 request dalam 10 detik dari IP sumber yang sama.

**Gambar 6. Simulasi Serangan DoS Menggunakan Perkakas Siege**

Wazuh SIEM berhasil mendeteksi lonjakan tersebut dan meneruskan alert melalui webhook ke API Gateway, yang selanjutnya mengirimkan notifikasi interaktif ke WhatsApp administrator berisi deskripsi ancaman, tingkat keparahan (Severity Level 12), stempel waktu, serta alamat IP penyerang (192.168.1.17) beserta dua opsi balasan. Pada pengujian pertama, administrator memilih opsi 2 (abaikan), sehingga sistem SOAR tidak mengeksekusi tindakan apa pun dan hanya mengirimkan konfirmasi bahwa alert telah diabaikan, membuktikan bahwa sistem tidak melakukan tindakan otomatis tanpa persetujuan eksplisit dari manusia.

Pada pengujian kedua, administrator membalas dengan opsi 1 (blokir), sehingga sistem terlebih dahulu mengirimkan pesan interim "Memproses Pemblokiran..." sebagai umpan balik real-time, kemudian mengeksekusi perintah `New-NetFirewallRule` pada Windows Defender Firewall untuk memblokir IP 192.168.1.17. Setelah perintah berhasil dijalankan, bot WhatsApp mengirimkan konfirmasi akhir berstatus "PEMBLOKIRAN BERHASIL" beserta rincian IP terblokir dan stempel waktu penyelesaian, sebagaimana ditunjukkan pada Gambar 7.

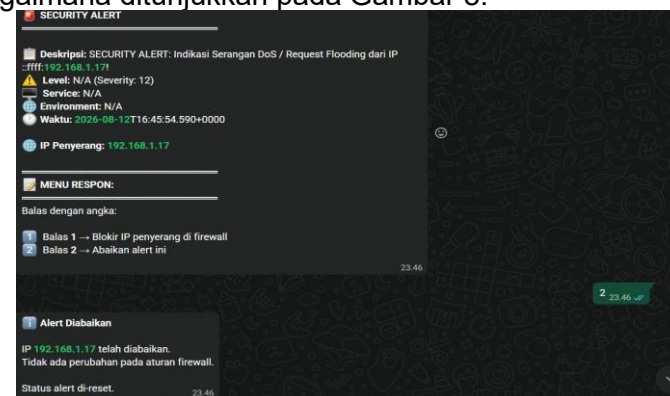


**Gambar 7. Eksekusi Respons Otomatis Pemblokiran IP Penyerang via WhatsApp Interaktif**

Verifikasi pasca-mitigasi menunjukkan bahwa akses dari mesin Kali Linux ke server target mengalami connection timed out, dan pengecekan melalui perintah `Get-NetFirewallRule` pada PowerShell mengonfirmasi terbentuknya aturan `Block_Attacker_192.168.1.17` dengan status `Enabled=True` dan `Action=Block`. Validasi silang antara notifikasi aplikasi dan kondisi riil sistem operasi ini membuktikan bahwa mekanisme active response bekerja secara nyata pada lapisan jaringan, bukan sekadar notifikasi semu.

### Pengujian Brute Force

Pengujian Brute Force dilakukan menggunakan skrip yang menyasar endpoint `http://192.168.91.10:3001/api/login` melalui metode HTTP POST, dengan kombinasi 15 username dan 8 password (120 kombinasi kredensial) untuk merepresentasikan skenario dictionary attack (Aji et al., 2025). Setiap percobaan dengan kredensial tidak valid direspons dengan HTTP 401 Unauthorized, dan setiap kegagalan dicatat sebagai log JSON dengan atribut `action: LOGIN_FAILED` beserta metadata endpoint, email, dan IP sumber. Akumulasi kegagalan login dari IP yang sama dalam waktu singkat berhasil memenuhi ambang batas frekuensi pada Rule ID 100011 (lima kegagalan dalam 60 detik), sehingga Wazuh SIEM mengidentifikasi pola tersebut sebagai anomali Brute Force dan memicu notifikasi WhatsApp secara instan, sebagaimana ditunjukkan pada Gambar 8.



**Gambar 8. Notifikasi Security Alert WhatsApp pada Skenario Brute Force**

Administrator membalas notifikasi dengan opsi 1, sehingga sistem SOAR mengeksekusi active response untuk memblokir IP 192.168.91.30 pada firewall, dan bot WhatsApp mengonfirmasi status "PEMBLOKIRAN BERHASIL" beserta stempel waktu penyelesaian. Hasil ini menunjukkan bahwa efektivitas deteksi Brute Force sangat bergantung pada kelengkapan struktur log yang dihasilkan aplikasi, sehingga keberhasilan deteksi tidak semata-mata bergantung pada kemampuan Wazuh sebagai mesin analisis, melainkan juga pada kualitas desain logging pada sisi aplikasi, yang menegaskan pentingnya prinsip security by design.

### Analisis Keterkaitan Hasil Pengujian

Apabila hasil pengujian white-box dan pengujian serangan dikaji secara terintegrasi, terlihat keterkaitan langsung antara jalur logika yang telah tervalidasi secara kode dengan perilaku aktual sistem saat menghadapi serangan nyata. Jalur pada handler WhatsApp yang memvalidasi respons "PEMBLOKIRAN BERHASIL" dan "Alert Diabaikan" secara empiris teramati kembali pada pengujian DoS maupun Brute Force ketika administrator memilih opsi 1 maupun opsi 2. Konsistensi ini membuktikan bahwa pengujian white-box pada tingkat kode program secara valid merepresentasikan perilaku sistem pada tingkat operasional. Perbandingan antara hasil pengujian DoS dan Brute Force juga memperlihatkan bahwa arsitektur SOAR yang dibangun bersifat generik (rule-agnostic), karena alur notifikasi interaktif dan mekanisme active response yang sama dapat digunakan untuk menangani lebih dari satu jenis insiden keamanan tanpa memerlukan modifikasi khusus pada logika SOAR Gateway, sehingga menunjukkan tingkat reusabilitas dan skalabilitas yang baik pada desain integrasi SIEM-SOAR yang dikembangkan. Secara keseluruhan, hasil pengujian membuktikan bahwa integrasi Wazuh SIEM dan sistem SOAR berbasis WhatsApp Gateway telah memenuhi tiga aspek utama, yaitu korektnes logika program pada tingkat kode, kemampuan deteksi anomali berbasis rule threshold untuk dua jenis serangan yang berbeda karakteristik, serta efektivitas mekanisme active response yang tervalidasi baik pada tingkat notifikasi aplikasi maupun pada tingkat konfigurasi sistem operasi target, sehingga mendukung tercapainya tujuan penelitian dalam mempercepat Mean Time to Respond (MTTR) penanganan insiden keamanan siber melalui pendekatan human-in-the-loop.

Meskipun hasil pengujian menunjukkan tingkat keberhasilan yang tinggi, penelitian ini memiliki beberapa keterbatasan yang perlu dicermati. Pertama, seluruh pengujian dilakukan pada lingkungan laboratorium terkontrol menggunakan aplikasi e-commerce dummy dan jaringan virtual terisolasi, sehingga karakteristik lalu lintas data belum sepenuhnya merepresentasikan kondisi produksi dengan volume transaksi yang jauh lebih besar dan beragam. Kedua, notifikasi dan eksekusi mitigasi masih bergantung pada satu nomor administrator (single admin), sehingga apabila administrator tidak responsif dalam jangka waktu tertentu, proses human-in-the-loop dapat menunda penanganan insiden. Ketiga, cakupan pengujian serangan pada penelitian ini masih terbatas pada dua jenis ancaman, yaitu Denial of Service pada Application Layer dan Brute Force pada endpoint otentikasi, sehingga efektivitas ruleset terhadap vektor serangan lain seperti SQL Injection maupun Cross-Site Scripting belum dapat dipastikan. Keterbatasan-keterbatasan ini menjadi acuan penting bagi arah pengembangan sistem pada penelitian selanjutnya.

### SIMPULAN

Wazuh SIEM berhasil diintegrasikan dengan API Gateway berbasis Node.js (whatsapp-web.js) untuk mendeteksi serangan DoS dan Brute Force pada server e-commerce secara real-time dan mengirimkan notifikasi ke WhatsApp administrator. Mekanisme mitigasi interaktif dua arah (human-in-the-loop) terbukti efektif menekan risiko false positive karena pemblokiran IP hanya dieksekusi setelah persetujuan administrator. Pengujian White-Box pada tiga fungsi kritis menunjukkan 19 dari 19 basis path berjalan sesuai ekspektasi (100%). Disarankan pengembangan lanjutan berupa notifikasi multi-admin, perluasan skenario serangan (DDoS, SQL Injection, XSS), fitur unblock IP otomatis, dan migrasi ke WhatsApp Business Cloud API resmi untuk meningkatkan keandalan sesi jangka panjang.

### DAFTAR PUSTAKA

- Aji, N. D., Sujaka, T. T., Asroni, O., & Latif, K. A. (2025). Analisis Dan Implementasi Algoritma Bcrypt Dengan Affine Cipher Untuk Pengamanan Password Pada Aplikasi Web Analysis And Implementation Of Bcrypt Algorithm With Affine Cipher For Password Security In Web Applications. 8(1), 1–9.
- Alamin, Z., & Amirul, M. (2025). Analisis Keamanan Jaringan Pada Sistem Kendali Jarak Jauh Untuk Infrastruktur Kritis Network Security Analysis Of Remote Control System For Critical Infrastructure. 1, 25–41. <https://doi.org/10.63866/Jpst.V1i1.39>
- Coppolino, L., Sgaglione, L., Antonio, S. D., Magliulo, M., Romano, L., & Pacelli, R. (2022).

- Risk Assessment Driven Use Of Advanced SIEM Technology For Cyber Protection Of Critical E - Health Processes. SN Computer Science, 3(1), 1–13. <https://doi.org/10.1007/S42979-021-00858-4>
- Fachri, F. (2023). Optimasi Keamanan Web Server Terhadap Serangan Brute-Force Menggunakan Penetration Testing. 10(1), 51–58. <https://doi.org/10.25126/Jtiik.2023105872>
- Kamil, I., & Miranda, T. (2024). Literature Review Pengaruh E-Commerce Terhadap UMKM Pada Era New Normal. 1(1), 35–43.
- Kouari, O. El, Lazaar, S., & Achoughi, T. (2025). Fortifying Industrial Cybersecurity : A Novel Industrial Internet Of Things Architecture Enhanced By Honeypot Integration. 15(1), 1089–1098. <https://doi.org/10.11591/Ijece.V15i1.Pp1089-1098>
- Kurnia, R., Widyatama, F., Wibawa, I. M., & Brata, Z. A. (2025). Enhancing Security Operations Center : Wazuh Security Event Response With Retrieval-Augmented-Generation-Driven Copilot. 1–26.
- Nedyalkov, I. (2024). Studying The Impact Of Different TCP Dos Attacks On The Parameters Of Voip Streams. 556–587.
- Ozkan-Okay, M., Yilmaz, A. A., & Akin, E. (2023). A Comprehensive Review Of Cyber Security Vulnerabilities ,.
- Padmavathi, V., & Saminathan, R. (2025). OPEN A Federated Edge Intelligence Framework With Trust Based Access Control For Secure And Privacy Preserving Iot Systems. 1–16.
- Putra, D. T., Wahyudi, I., Megavitry, R., & Supriadi, A. (N.D.). Pemanfaatan E-Commerce Dalam Pemasaran Hasil Pertanian : Kelebihan Dan Tantangan Di Era Digital. 02(08).
- Putra, F. P. E., Khairi, M., Hidayatullah, M. I., & Maulana, I. (2025). Analisis Protokol Keamanan Jaringan Dalam Era Internet Of Things (Iot). 8(2), 356–366.
- Saputra, F. A., Dharmawan, T. R., & Rustianto, A. (2024). Implementasi Wazuh Siem Untuk Manajemen Log Event Di Pesantren Teknologi Informasi Dan Komunikasi Jombang. 10(2), 146–155.
- Shafiyah, A., Nama, G. F., & Pradipta, R. A. (2024). Implementasi Wazuh Menggunakan Metode PPDIOO Di Sistem Keamanan Jaringan PSDKU Universitas Lampung Waykanan Sebagai Deteksi Dan Respon Serangan Siber. 12(2).
- Sie, J. B. L., Musdar, I. A., Bahri, S., & Informatika, T. (2022). Pengujian White Box Testing Terhadap Website Room Menggunakan Teknik Basis Path. 02, 45–57.
- Syaikhurrahman, M., & Prasetyo, B. (2025). Deteksi Serangan DDOS Menggunakan Deep Learning Dalam Administrasi Jaringan. 6(6), 8995–9003. <https://doi.org/10.54373/Imeij.V6i6.4112>
- Tafonao, F., & Raslina, N. N. (2025). Perancangan Website E-Commerce Untuk Fayucom Dengan Integrasi Payment Gateway Xendit Menggunakan Html , Css , Dan Javascript. 5(8), 3203–3214.
- Zahid, H., Hina, S., Hayat, M. F., & Shah, G. A. (2023). Agentless Approach For Security Information And Event Management In Industrial Iot.