Department of Digital Business
Journal of Artificial Intelligence and Digital Business (RIGGS)
Homepage: https://journal.ilmudata.co.id/index.php/RIGGS
Vol. 5 No. 2 (2026) pp: 20024-20035
P-ISSN: 2963-9298, e-ISSN: 2963-914X

Desain dan Implementasi *Wazuh Active Response* untuk Mitigasi Serangan Konten Judi *Online*

Gian Virgiawan[1], I Putu Hariyadi[2]
[12]Ilmu Komputer, Fakultas Teknik, Universitas Bumigora
[1]2201010041@universitasbumigora.ac.id , [2]putu.hariyadi@universitasbumigora.ac.id

Abstrak

Serangan bermotif judi online yang menyasar situs pemerintah dan akademik semakin meningkat, di mana pelaku mengeksploitasi celah unggah file untuk menyisipkan gambar promosi judi, webshell, kata kunci, maupun tautan tersembunyi (SEO Poisoning). Sistem Security Information and Event Management (SIEM) seperti Wazuh telah banyak digunakan untuk mendeteksi ancaman ini, namun sebagian besar penelitian terdahulu masih bersifat pasif, hanya memberikan peringatan tanpa tindakan pemulihan otomatis, serta mengandalkan verifikasi berbasis hash yang tidak mampu mengenali konten judi yang disamarkan di dalam gambar atau modifikasi file yang bersifat zero-day. Penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem keamanan berlapis (Multi-Layer Defense) berbasis Wazuh yang mengintegrasikan File Integrity Monitoring (FIM), OCR berbasis Tesseract dan OpenCV, domain blacklist berisi 98.730 domain, deteksi webshell berbasis Regex, dan pemantauan syscall menggunakan Linux Auditd, yang dilengkapi fitur Active Response untuk meremediasi ancaman secara otomatis serta notifikasi real-time melalui Telegram Bot API. Metode penelitian yang digunakan adalah Network Development Life Cycle (NDLC) dengan tiga tahapan, yaitu analysis, design, dan simulation prototyping. Pengujian dilakukan menggunakan metode Black-Box Testing pada lima skenario serangan yang diulang lima kali pada tiga kondisi operasional berbeda. Hasil pengujian menunjukkan bahwa sistem berhasil mendeteksi dan memitigasi seluruh skenario serangan dengan tingkat keberhasilan 100% serta rata-rata waktu remediasi di bawah 15 detik, jauh lebih cepat dibandingkan proses penanganan manual.

***Kata Kunci**: Keamanan Siber, Wazuh, Active Response, Judi Online, SIEM, File Integrity Monitoring*

1. Latar Belakang

Seiring berkembangnya layanan digital pemerintah, banyak desa kini memiliki situs web resmi yang digunakan untuk menyampaikan informasi, melayani laporan warga, dan menyimpan data. Namun kemudahan akses internet ini juga membuka celah bagi kejahatan siber. Salah satu ancaman yang semakin marak adalah serangan bermotif judi *online*, di mana pelaku menargetkan situs pemerintah berakhiran .go.id untuk menyisipkan konten promosi perjudian [NO_PRINTED_FORM] [1]. Tujuan pelaku bukan sekadar merusak tampilan situs, tetapi juga memanfaatkan reputasi *domain* pemerintah untuk mendongkrak peringkat situs judi di mesin pencari melalui teknik yang dikenal sebagai *SEO Poisoning*. Ancaman siber semacam ini terus berkembang dari tahun ke tahun, menambah tantangan bagi organisasi dalam menjaga keamanan informasi dan infrastrukturnya [2].

Serangan ini umumnya dilakukan dengan mengeksploitasi fitur unggah *file* yang tersedia pada aplikasi web. Kerentanan pada fitur *upload file* memungkinkan penyerang menyisipkan file berbahaya tanpa terdeteksi oleh sistem keamanan konvensional [3]. Melalui celah tersebut, pelaku dapat mengunggah gambar berisi teks promosi judi, file berbahaya seperti *webshell* yang memungkinkan pelaku mengendalikan *server* dari jarak jauh, atau menyisipkan tautan judi tersembunyi langsung ke dalam *source code* halaman situs. Ketika serangan ini berhasil, dampaknya sangat serius: situs pemerintah dapat diblokir oleh mesin pencari, kepercayaan masyarakat terhadap institusi menurun, dan data warga terancam [4]. Penanganan serangan berlapis pada fitur unggah file juga menunjukkan bahwa pendekatan mitigasi tunggal tidak cukup untuk menanggulangi variasi serangan yang semakin beragam [5].

Untuk menghadapi ancaman ini, banyak instansi mengadopsi *Security Information and Event Management* (SIEM) sebagai solusi terpadu untuk mendeteksi, menganalisis, dan merespons ancaman siber secara lebih efisien (Khotimah et al., 2022). *Wazuh* adalah salah satu platform SIEM *open-source* yang banyak digunakan, mampu mengumpulkan dan menganalisis *log* dari *server* secara *real-time*, termasuk mendeteksi perubahan file melalui modul *File Integrity Monitoring* (FIM) [6]. Beberapa penelitian telah membuktikan efektivitasnya, di antaranya implementasi *Wazuh* untuk monitoring keamanan *server* [7] analisis sistem SIEM *Wazuh* pada instansi pemerintah daerah [8], serta penggunaan *Wazuh* sebagai *log event management* untuk mendeteksi serangan DoS [9]. Penelitian

lain juga membuktikan bahwa modul FIM pada *Wazuh* efektif mendeteksi serangan *defacement* dengan akurasi tinggi [10], termasuk dikombinasikan dengan Snort IDS berbasis *signature* untuk kasus judi *online* [4].

Untuk memperkuat kemampuan deteksi, beberapa penelitian mengembangkan integrasi *Wazuh* dengan sistem notifikasi. Penelitian terdahulu mengintegrasikan *Wazuh* dengan *WhatsApp Bot* untuk mempercepat penyampaian notifikasi kepada *administrator* [11], sementara penelitian lain mengintegrasikan *Wazuh* dengan *VirusTotal* untuk penghapusan malware otomatis [12]. Di sisi deteksi konten visual, pendekatan multimodal menggunakan *Faster R-CNN* dan *Tr-OCR* terbukti efektif mendeteksi iklan judi *online* yang disamarkan [13], membuktikan bahwa teknologi OCR memiliki potensi besar dalam mendeteksi konten judi yang tersembunyi di dalam gambar [14].

Namun, kelemahan mendasar yang masih ditemukan pada sebagian besar sistem tersebut adalah sifatnya yang pasif, sistem hanya mendeteksi dan memberikan peringatan, tetapi tidak mengambil tindakan apapun untuk membersihkan atau memulihkan file yang sudah terinfeksi [12]. Artinya, file berbahaya akan tetap aktif di *server* selama *administrator* belum sempat membuka *dashboard* dan merespons secara manual. Penelitian terkait implementasi *Active Response* pada *Wazuh* dan *Suricata* menunjukkan bahwa pendekatan respons otomatis mampu mereduksi waktu penanganan insiden secara signifikan [15], namun penerapannya pada kasus serangan judi *online* yang spesifik masih sangat terbatas.

Selain itu, sistem yang sudah menerapkan respons otomatis umumnya masih mengandalkan pencocokan *hash* atau *signature* untuk memverifikasi ancaman. Metode ini memiliki kelemahan fundamental yaitu file berbahaya yang baru (*zero-day*) atau gambar berisi teks judi tidak akan terdeteksi karena nilainya belum terdaftar di *database* keamanan manapun. Dari berbagai celah tersebut, penelitian ini mengusulkan sistem yang tidak hanya mendeteksi ancaman, tetapi juga langsung membersihkan atau memulihkan kondisi *server* secara otomatis melalui analisis berbasis isi file (*content-based verification*), dengan menerapkan kerangka keamanan berlapis yang selaras dengan standar NIST dan ISO/IEC 27001 [16].

2. Metode Penelitian

2.1 Metode Pengembangan

Penelitian ini menggunakan metode *Network Development Life Cycle* (NDLC), yaitu sebuah pendekatan pengembangan infrastruktur jaringan yang terstruktur dan bersifat siklus [17]. NDLC merupakan siklus proses perancangan infrastruktur jaringan yang memungkinkan terjadinya pemantauan untuk mengetahui statistik dan kinerja jaringan [18]. NDLC memiliki enam tahapan yaitu *analysis, design, simulation prototyping, implementation, monitoring*, dan *management*, dan penelitian ini hanya menggunakan tiga tahapan pertama yaitu *analysis, design, simulation prototyping*.

Untuk mendukung infrastruktur sistem, penelitian ini memanfaatkan teknologi kontainerisasi menggunakan *Docker* sebagai lingkungan terisolasi bagi aplikasi web yang diuji. Studi tentang keamanan internal *Docker* menunjukkan bahwa kontainerisasi memberikan lapisan isolasi yang efektif antara aplikasi dan sistem operasi *host*, sehingga eksploitasi yang berhasil pada *conatiner* tidak merambat ke sistem induk [19]. Selain itu, pemahaman mendalam tentang arsitektur jaringan dan protokol *TCP/IP* menjadi fondasi penting dalam merancang topologi jaringan yang aman dan handal [20], termasuk pemanfaatan teknologi virtualisasi berbasis kontainer sebagai infrastruktur pendukung sistem [21].

Dari sisi kerangka keamanan, penelitian ini menerapkan prinsip *defense-in-depth* yang sejalan dengan standar keamanan informasi NIST dan ISO/IEC 27001, di mana perlindungan berlapis dirancang untuk memastikan bahwa kegagalan satu lapisan tidak menyebabkan kompromi sistem secara keseluruhan [16]. Pendekatan implementasi *Wazuh* menggunakan metode terstruktur juga menjadi rujukan dalam merancang alur konfigurasi sistem pada penelitian ini [22].

1. Analysis

Pada tahap ini, peneliti mengumpulkan data dan informasi melalui studi literatur untuk memahami pola serangan judi *online* yang menyasar situs web pemerintah, serta mempelajari teknologi yang relevan. Studi literatur mencakup jurnal ilmiah, skripsi, dan dokumentasi resmi terkait *Wazuh* SIEM, teknik *File Integrity Monitoring*, *Optical Character Recognition* (OCR), serta jenis-jenis serangan file upload seperti *webshell* dan *SEO Poisoning*. Hasil analisis ini digunakan sebagai dasar untuk merancang solusi yang tepat.

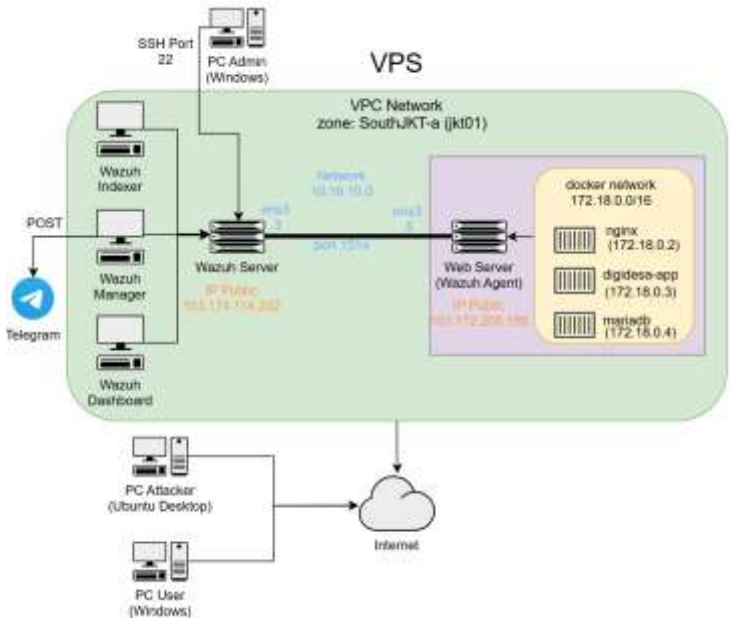
2. *Design*

Pada tahap ini, peneliti merancang cetak biru sistem keamanan yang akan dibangun berdasarkan temuan pada tahap analisis.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

A. Rancangan Topologi Jaringan Uji Coba

Rancangan uji coba disimulasikan menggunakan dua *Virtual Private Server* (VPS) bersistem operasi *Ubuntu Server* 24.04 LTS. VPS pertama difungsikan sebagai *Wazuh Server* yang terinstal *Wazuh Manager* dan *Wazuh Indexer*, sedangkan VPS kedua difungsikan sebagai Web *Server* yang menjalankan *Wazuh Agent* dan menjadi *host Docker* bagi aplikasi web *Digidesa*, yang terdiri atas tiga kontainer yaitu *Nginx*, PHP-FPM, dan MariaDB 10.11. Pemisahan kedua server bertujuan agar *Wazuh Server* tetap berjalan aman meskipun Web *Server* berhasil dilumpuhkan oleh penyerang. Rancangan topologi jaringan uji coba dapat dilihat pada gambar 1.



Gambar 1 Rancangan Topologi Jaringan Uji Coba

B. Rancangan Pengalamatan IP

Untuk pengalamatan IP pada rancangan uji coba ini digunakan kombinasi alamat IP publik dan alamat IP privat. IP *private* menggunakan network 10.10.10.0/24 untuk jalur komunikasi lokal antar *server* di lingkungan VPS, sedangkan IP publik digunakan untuk akses dari luar, dan *network* 172.18.0.0/16 dialokasikan khusus untuk mengisolasi layanan *container Docker*. Detail alokasi pengalamatan IP pada masing-masing perangkat dapat dilihat pada tabel 1.

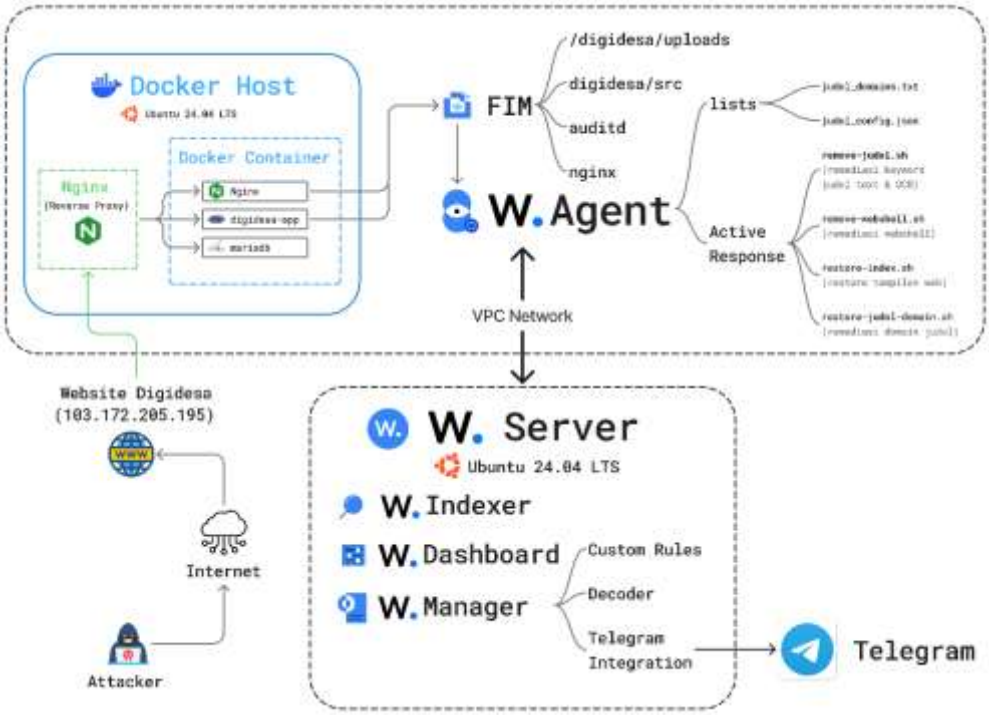
Tabel 1 Pengalamatan IP

No	Nama Perangkat	IP Publik	IP Private	Subnet Mask
1	Wazuh Server	103.174.114.202	10.10.10.3	/24
2	Wazuh Agent	103.172.205.195	10.10.10.5	/24
3	Docker Host Network	-	172.18.0.1	/16
4	Container Nginx	-	172.18.0.2	/16
5	Container PHP-FPM	-	172.18.0.3	/16
6	Container MariaDB	-	172.18.0.4	/16

C. Rancangan Alur Kerja Sistem

Alur kerja sistem dirancang agar mampu beroperasi secara otomatis tanpa bergantung penuh pada *administrator*, dengan menerapkan pendekatan keamanan berlapis (*Multi-Layer Defense*) yang terdiri dari lima lapisan deteksi. Rancangan alur kerja sistem dapat dilihat pada gambar 2.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)



Gambar 2 Rancangan Alur Kerja Sistem

Kelima lapisan deteksi tersebut meliputi FIM berbasis *inotify kernel Linux* untuk memantau direktori `/uploads/` dan `/src/`, OCR berbasis *Tesseract* dan *OpenCV* untuk mengekstrak teks dari gambar yang diunggah, *domain blacklist* berisi 98.730 *domain* yang tersimpan dalam struktur data *hash set* untuk pencarian dengan kompleksitas O(1), deteksi *webshell* berbasis *Regex* pada ekstensi file di direktori *uploads*, dan pemantauan *syscall* menggunakan *Linux Auditd*. Ketika salah satu lapisan menemukan ancaman, *Wazuh Manager* memerintahkan *Wazuh Agent* menjalankan *script Active Response* yang sesuai, meliputi remove-judol.sh, remove-webshell.sh, restore-index.sh, dan restore-judol-domain.sh, yang masing-masing bekerja tanpa menunggu *administrator*.

D. Kebutuhan Perangkat Keras dan Perangkat Lunak

Kebutuhan perangkat keras pada penelitian ini terdiri dari dua server VPS, yaitu Web *Server* (*Wazuh Agent*) dengan spesifikasi 2 vCPU, RAM 2 GB, penyimpanan 20 GB, dan *Wazuh Server* dengan spesifikasi 8 vCPU, RAM 16 GB, penyimpanan 120 GB. Adapun perangkat lunak yang digunakan meliputi *Wazuh* versi 4.14.4, *Ubuntu* 24.04 LTS, *Docker Engine* dan *Docker Compose*, *Nginx*, PHP-FPM, *MariaDB* 10.11, *Python* 3, *OpenCV*, *PyTesseract* dan *Tesseract* OCR (ind+eng), *Linux Auditd*, serta *Telegram Bot* API.

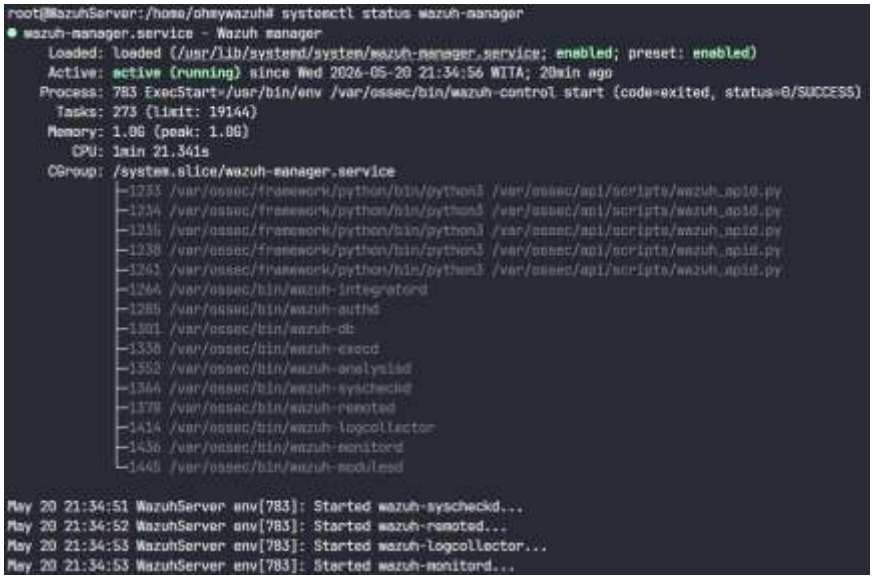
3. *Simulation Prototyping*

Pada tahap ini, seluruh rancangan diimplementasikan ke dalam lingkungan VPS, mencakup instalasi dan konfigurasi semua komponen pendukung, kemudian dilakukan pengujian *Black-Box Testing* menggunakan lima skenario serangan yang masing-masing diulang lima kali pada setiap kondisi.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

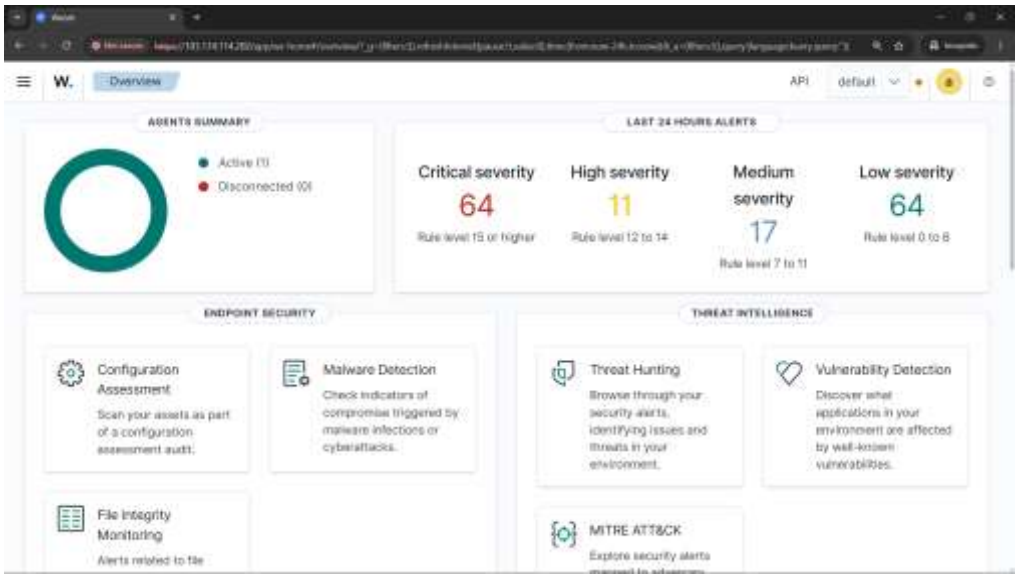
3. Hasil dan Pembahasan

3.1 Hasil Instalasi *Wazuh Server*



Gambar 3 Verifikasi instalasi wazuh manager

Pada gambar 3 terlihat bahwa status *Wazuh Manager* menunjukkan kondisi aktif (*running*), yang berarti komponen utama sistem keamanan telah berjalan dengan baik pada *Wazuh Server* dan siap memantau *Wazuh Agent*.

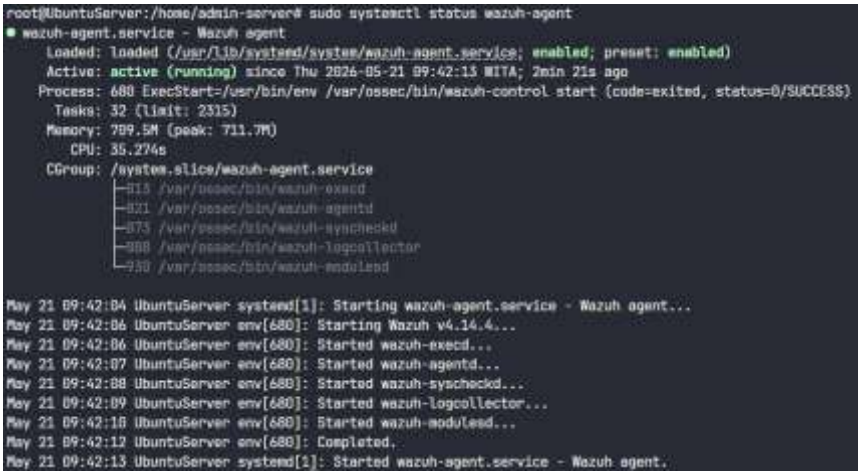


Gambar 4 Tampilan wazuh dashboard

Pada gambar 4 terlihat tampilan *Wazuh Dashboard* yang berfungsi sebagai pusat pemantauan seluruh aktivitas keamanan. Melalui *dashboard* ini, seluruh *alert* dan status *agent* dapat dipantau secara terpusat.

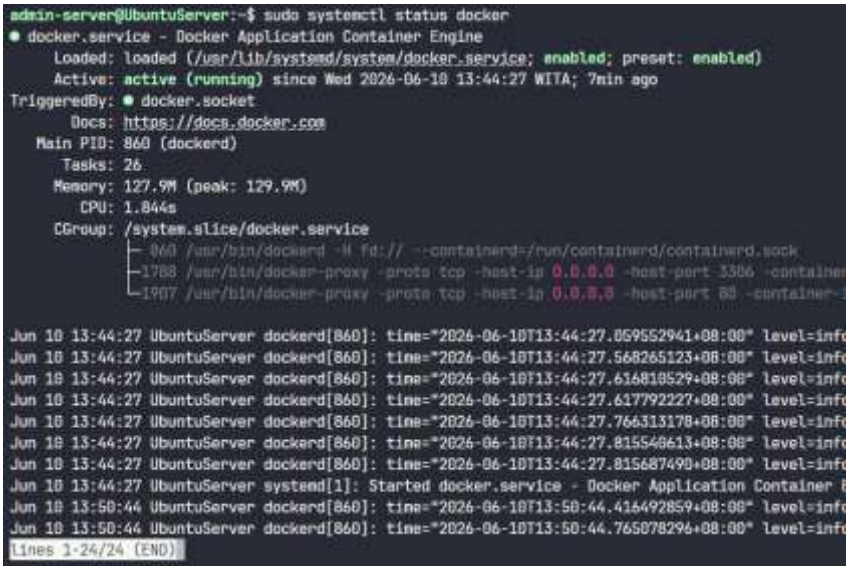
DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

3.2 Hasil Instalasi *Wazuh Agent*



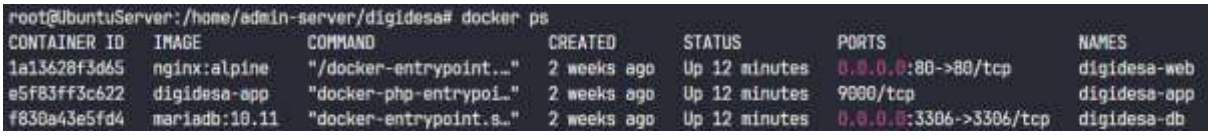
Gambar 5 Verifikasi instalasi wazuh agent

Pada gambar 5 merupakan hasil instalasi *Wazuh Agent* yang digunakan untuk memantau aktifitas yang ada pada web *server*.



Gambar 6 Verifikasi instalasi docker engine

Pada gambar 6 merupakan hasil instalasi *Docker Engine* dan *Docker Compose* yang digunakan untuk menjalankan kontainer *Nginx*, PHP-FPM, dan *MariaDB* pada *Wazuh Agent*.



Gambar 7 Verifikasi app docker container

Pada gambar 7 terlihat bahwa aplikasi web *Digidesa* berhasil berjalan di atas tiga *conatiner Docker* dan dapat diakses melalui browser, menandakan lingkungan pengujian telah siap digunakan.

3.3 Hasil Konfigurasi *File Integrity Monitoring* (FIM)

Agar *Wazuh* dapat mendeteksi ancaman dan melakukan remediasi, modul *File Integrity Monitoring* (FIM) diaktifkan *Wazuh Agent*. Atribut `realtime="yes"` ditanamkan pada direktori *source code* dan *uploads* untuk selalu merespon setiap terjadi perubahan pada *directory* atau *source code*, seperti terlihat pada gambar 8.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

```
<!-- File integrity monitoring -->
<syscheck>
  <disabled>no</disabled>

  <!-- Frequency that syscheck is executed default every 12 hours -->
  <frequency>43200</frequency>

  <scan_on_start>yes</scan_on_start>

  <!-- Directories to check  (perform all possible verifications) -->
  <directories>/etc,/usr/bin,/usr/sbin</directories>
  <directories>/bin,/sbin,/boot</directories>

  <!-- admin server dir -->
  <directories check_all="yes" report_changes="yes" realtime="yes">/home/admin-server</directories>
  <directories check_all="yes" report_changes="yes" realtime="yes">/etc/passwd</directories>

  <!-- Digidesa -->
  <directories realtime="yes" check_all="yes" report_changes="yes">/home/admin-server/digidesa/src</directories>
  <directories realtime="yes" check_all="yes" report_changes="yes">/home/admin-server/digidesa/uploads</directories>
  <directories realtime="yes" check_all="yes">/home/admin-server/digidesa/docker-compose.yml</directories>
  <directories realtime="yes" check_all="yes">/home/admin-server/digidesa/nginx.conf</directories>
  <directories realtime="yes" check_all="yes">/home/admin-server/digidesa/Dockerfile.php</directories>
  <ignore>/home/admin-server/digidesa/mariadb_data</ignore>
  <ignore>/home/admin-server/digidesa/logs</ignore>
```

Gambar 8 Konfigurasi FIM

3.4 Hasil Konfigurasi *Custom Rules*

Aturan deteksi (*Rules*) dikonfigurasi pada `/var/ossec/etc/rules/local_rules.xml` di *server Wazuh Server*. Aturan ini membedah log masuk dan menerjemahkannya sebagai level ancaman.

```
<rule id="500550" level="15">
  <if_sid>550</if_sid>
  <field name="changed_content" type="pcre2">(?i)slot|gacor|maxwin|thailand|sigmaslot|zeus|cuan</field>
  <description>[Judol File Modification]: File $(file) contains a judol signature</description>
  <group>syscheck_entry_modified,syscheck_file,judol_file</group>
</rule>
```

Gambar 9 Konfigurasi rule 500550

Gambar 9 menunjukkan *Rule* 500550 (*Level* 15) yang memindai isi teks file menggunakan PCRE2 untuk mendeteksi kata kunci judi seperti "slot", "gacor", dan "maxwin", menangani serangan unggahan file teks berisi promosi judi dan web *defacement*.

```
<rule id="500554" level="12">
  <if_sid>554</if_sid>
  <field name="file" type="pcre2">(?i)public|www|html|digidesa|src|uploads</field>
  <description>[File creation on web directory]: New file $(file) on web directory</description>
  <group>syscheck_entry_added,syscheck_file</group>
</rule>
```

Gambar 10 Konfigurasi rule 500554

Gambar 10 menunjukkan *Rule* 500554 (*Level* 15) yang memicu proses OCR setiap ada file gambar baru di direktori *uploads* untuk mendeteksi teks promosi judi yang tersembunyi di dalam gambar.

```
<rule id="500551" level="15">
  <if_sid>550</if_sid>
  <description>[SEO Poisoning]: Perubahan file $(file) perlu dicek domain blacklist</description>
  <group>syscheck_entry_modified,syscheck_file,judol_file</group>
</rule>
```

Gambar 11 Konfigurasi rule 500551

Gambar 11 menunjukkan *Rule* 500551 (*Level* 15) yang mencocokkan setiap tautan di dalam file terhadap 98.730 domain judi, menangani serangan *SEO Poisoning* meskipun tidak mengandung kata kunci judi.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

```
<rule id="100522" level="15">
  <if_sid>500554</if_sid>
  <field name="file" type="pcre2">(?i)uploads/.*\.(php|phtml|php5|phar)$</field>
  <description>[WebShell Upload]: File PHP berbahaya diunggah ke direktori uploads: $(file)</description>
  <group>security_event,webshell</group>
</rule>
```

Gambar 12 Konfigurasi rule 100522

Gambar 12 menunjukkan *Rule* 100522 (Level 12) yang mendeteksi file berekstensi PHP di direktori uploads, menangani serangan *webshell* dan *backdoor*.

```
<rule id="100030" level="12">
  <decoded_as>judol_cleanup</decoded_as>
  <description>Active Response: File $(target_file) dimusnahkan karena m
  <group>syscheck_cleanup,security_event</group>
</rule>
```

Gambar 13 Konfigurasi rule 100030

Gambar 13 menunjukkan *Rule* 100030 yang menangkap log hasil eksekusi *Active Response* dan meneruskannya sebagai notifikasi ke *Telegram administrator*.

3.5 Hasil Konfigurasi *Active Response*

```
<active-response>
  <command>restore-judol-domain</command>
  <location>local</location>
  <rules_id>500551</rules_id>
</active-response>

<active-response>
  <command>restore-index</command>
  <location>local</location>
  <rules_id>500550</rules_id>
</active-response>

<active-response>
  <command>remove-judol</command>
  <location>local</location>
  <rules_id>500550,500554</rules_id>
</active-response>

<active-response>
  <command>remove-webshell</command>
  <location>local</location>
  <rules_id>100522</rules_id>
</active-response>
```

Gambar 14 Konfigurasi active response

Pada gambar 14 terlihat konfigurasi *Active Response* pada file ossec.conf *Wazuh Manager*. Konfigurasi ini menghubungkan setiap *rule* yang terpicu dengan *script custom rules* yang sesuai, sehingga begitu ancaman terdeteksi, perintah remediasi langsung dikirimkan ke *Wazuh Agent* secara otomatis.

3.6 Hasil Konfigurasi Notifikasi *Telegram*

```
<!-- Telegram integration -->
<integration>
  <name>custom-telegram.py</name>
  <rule_id>500554,500550,100030,100521,100522</rule_id>
  <alert_format>json</alert_format>
</integration>
```

Gambar 15 Konfigurasi custom-telegram

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

Pada gambar 15 terlihat konfigurasi integrasi *Telegram Bot* API yang memungkinkan sistem mengirimkan laporan insiden secara *real-time* ke *administrator*. Laporan yang dikirimkan memuat informasi lengkap, mencakup jenis ancaman yang ditemukan, nama file berbahaya, kata kunci atau domain judi yang terdeteksi, dan lapisan deteksi yang bekerja.

3.7 Hasil Uji Coba

Untuk mendapatkan tolok ukur perbandingan yang ilmiah, seluruh 5 skenario serangan (*Upload* gambar judol, u*pload webshell*, *upload* teks PHP judol, *defacement*, *SEO Poisoning*) diujikan masing-masing 5 kali repetisi, dan pada 3 kondisi yang berbeda yaitu:

a. Kondisi A: *Server* beroperasi standar tanpa perlindungan *Wazuh* SIEM.
b. Kondisi B: *Server* menggunakan *Wazuh* SIEM, tanpa mengaktifkan fitur *Active Response.*
c. Kondisi C: *Server* menggunakan *Wazuh* SIEM dengan *Active Response*.

A. Skenario *Upload* Gambar Promosi Judi *Online*

t	agent.name	UbuntuServer
t	data.detection_layer	LAYER-2 (OCR)
t	data.judol_keyword	toto
t	data.target_file	/home/admin-server/digidesa/uploads/citizen_reports/83_fawaz_2026-05-15/judol1.png
t	decoder.name	judol_cleanup
t	full_log	May 15 01:08:26 UbuntuServer wazuh-ar[7304]: TINDAKAN: Penghapusan file /home/admin-server/digidesa/uploads/citizen_reports/83_fawaz_2026-05-15/judol1.png. ALASAN: Terdeteksi kata kunci 'toto' via LAYER-2 (OCR).

Gambar 16 Log skenario upload gambar judol

Pada gambar 16 terlihat log *Wazuh* yang mencatat kejadian saat penyerang mengunggah gambar berisi teks promosi judi ke *direktori uploads*. Modul FIM mendeteksi keberadaan file baru dan melaporkannya ke *Wazuh Manager*. *Wazuh Manager* memicu *rule* 500554, kemudian memerintahkan *script* remove-judol.sh untuk menjalankan *OpenCV* dan *Tesseract* OCR guna mengekstrak teks dari gambar. Begitu kata kunci judi ditemukan di dalam gambar, file langsung dihapus secara otomatis.

B. Skenario *Upload Webshell* atau *Backdoor*

t	agent.name	UbuntuServer
t	decoder.name	judol_cleanup
t	full_log	May 15 01:15:02 UbuntuServer wazuh-ar[7357]: TINDAKAN: Penghapusan web shell /home/admin-server/digidesa/uploads/citizen_reports/88_fawaz_2026-05-15/shell.php. ALASAN: File PHP terdeteksi di direktori uploads.

Gambar 17 Log skenario upload webshell

Pada gambar 17 terlihat log *Wazuh* yang mencatat kejadian saat penyerang mengunggah file PHP berbahaya ke *direktori uploads*. *Wazuh Manager* memicu *rule* 100522 untuk mendeteksi keberadaan file berekstensi PHP di dalam folder yang seharusnya hanya berisi file media. *Script* remove-webshell.sh langsung dijalankan dan menghapus file tersebut secara langsung tanpa perlu membaca isi file.

C. Skenario *Upload* File PHP Berisi Kata Kunci Judi *Online*

t	agent.name	UbuntuServer
t	decoder.name	judol_cleanup
t	full_log	May 15 01:22:00 UbuntuServer wazuh-ar[7398]: TINDAKAN: Penghapusan web shell /home/admin-server/digidesa/uploads/citizen_reports/93_fawaz_2026-05-15/judol.php. ALASAN: File PHP terdeteksi di direktori uploads.

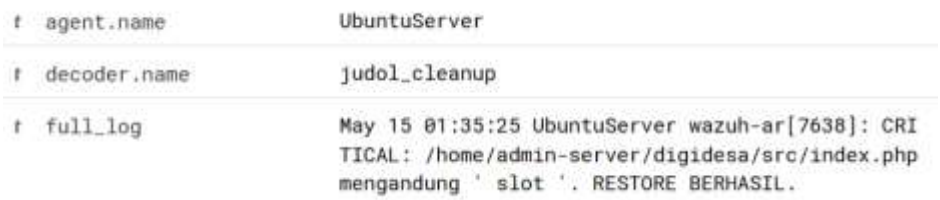
Gambar 18 Log skenario upload file php konten judol

Pada gambar 18 terlihat log *Wazuh* yang mencatat kejadian saat penyerang mengunggah file teks atau PHP yang isinya penuh dengan kata-kata promosi judi. *Wazuh Manager* memicu *rule* 500550, kemudian *script* remove-

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

judol.sh membaca isi teks file tersebut dan mencocokkannya dengan daftar kata kunci di judol_config.json. Begitu kecocokan (*strings matching*) ditemukan, file langsung dihapus.

D. Skenario Web *Defacement*



Gambar 19 Log skenario web defacement

Pada gambar 19 terlihat log *Wazuh* yang mencatat kejadian saat penyerang memodifikasi paksa file index.php dan menyisipkan konten promosi judi di halaman utama. FIM mendeteksi perubahan pada file index.php dan melaporkannya ke *Wazuh Manager*. *Rule* 500550 terpicu, kemudian *script* restore-index.sh membaca isi index.php yang telah diubah dan mencocokkan teksnya dengan daftar kata kunci judi. Begitu kecocokan ditemukan, file index.php langsung dipulihkan dari *direktori* cadangan di `/opt/wazuh-backup/src/` sehingga tampilan web kembali normal secara otomatis atau *self-healing*.

E. Skenario SEO *Poisoning Hidden Link*



Gambar 20 Log skenario SEO Poisoning

Pada gambar 20 terlihat log *Wazuh* yang mencatat kejadian saat penyerang menyisipkan tautan judi tersembunyi langsung ke dalam kode index.php tanpa menggunakan kata kunci judi yang jelas. FIM mendeteksi perubahan pada file index.php dan melaporkannya ke *Wazuh Manager*. *Rule* 500551 terpicu, kemudian *script* restore-judol-domain.sh mengekstrak semua tautan yang ada di dalam file menggunakan *Regex* dan mencocokkannya terhadap 98.730 *domain* judi yang tersimpan dalam struktur data *hash set*. Kecocokan *domain* ditemukan, dan file index.php langsung dipulihkan ke versi aman dari direktori Cadangan.

3.8 Hasil Analisa

Hasil uji coba yang telah dilakukan dipaparkan pada tabel berikut dengan parameter *detection rate* dan *mitigation rate* pada masing-masing kondisi, kemudian tabel berikutnya berupa rekapitulasi waktu remediasi pada Kondisi C.

Tabel 2 Hasil Pengujian Detection Rate dan Mitigation Rate

Skenario Serangan	**Kondisi A (Tanpa SIEM)**	**Kondisi B (Wazuh SIEM)**	**Kondisi C (Wazuh SIEM dan Active Response)**
Upload Gambar Promosi Judol	Tidak Terdeteksi	Terdeteksi (100%)	Deteksi, Remediasi (100%)
Upload Webshell/Backdoor	Tidak Terdeteksi	Terdeteksi (100%)	Deteksi, Remediasi (100%)
Upload File PHP *Keyword* Judol	Tidak Terdeteksi	Terdeteksi (100%)	Deteksi, Remediasi (100%)
Defacement index.php	Tidak Terdeteksi	Terdeteksi (100%)	Deteksi, Remediasi (100%)
SEO Poisoning Hidden Link	Tidak Terdeteksi	Terdeteksi (100%)	Deteksi, Remediasi (100%)

Pada tabel 2, pada kondisi A, seluruh serangan tidak terdeteksi sama sekali karena tidak ada mekanisme pemantauan, sehingga file berbahaya dapat tersimpan dan dieksploitasi. Pada Kondisi B, seluruh serangan berhasil terdeteksi oleh *Wazuh* dan memunculkan *alert* pada *dashboard*, namun file berbahaya tetap ada di *server* karena tidak ada tindakan pembersihan otomatis. Pada Kondisi C, sistem berhasil mendeteksi sekaligus memitigasi seluruh ancaman secara otomatis tanpa memerlukan bantuan *administrator*, dengan tingkat keberhasilan 100% pada semua skenario.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

Tabel 3 Rekapitulasi Waktu Remediasi pada Kondisi C

Skenario Serangan	Rata-rata Ttotal (detik)	Waktu Tercepat	Waktu Terlama
Upload Gambar Promosi Judol (OCR)	6,22	4,04	12,28
Upload Webshell/Backdoor (*Regex*)	1,00	0,08	2,09
Upload File PHP *Keyword* Judol (Teks)	1,38	0,79	1,7
Defacement index.php (*Auto-Restore*)	14,53	0,2	70,01
SEO Poisoning Hidden Link (Domain)	1,23	0,04	1,99

Pada tabel 3, terdapat beberapa temuan penting yang perlu dibahas. Skenario *webshell* mencatatkan waktu remediasi rata-rata tercepat yaitu 1,00 detik karena sistem tidak perlu membaca isi file, cukup memeriksa ekstensinya saja. Skenario *SEO Poisoning* juga menunjukkan kecepatan yang bagus, yaitu rata-rata 1,23 detik meskipun harus mencocokkan tautan dengan hampir seratus ribu *domain blacklist*, hal ini membuktikan bahwa penggunaan struktur data *hash set* menghasilkan kecepatan pencarian yang konstan tanpa terpengaruh oleh jumlah data. Skenario deteksi gambar judi membutuhkan waktu rata-rata lebih lama yaitu 6,22 detik karena adanya proses komputasi tambahan berupa konversi gambar dan ekstraksi teks oleh mesin OCR. Skenario *defacement* mencatat rata-rata tertinggi yaitu 14,53 detik, yang sebagian besar disebabkan oleh lonjakan waktu pada percobaan pertama (70,01 detik) akibat anomaly yaitu *cold start*, yaitu kondisi di mana sistem perlu memetakan *direktori* cadangan untuk pertama kalinya setelah *server* dinyalakan; pada percobaan berikutnya, waktu pemulihan turun drastis ke angka di bawah 2 detik karena data cadangan telah tersimpan di memori cache.

Secara keseluruhan, sistem yang dibangun berhasil membuktikan tiga keunggulan utama dibandingkan sistem terdahulu. Pertama, dari sisi kecepatan, pekerjaan remediasi yang secara manual dapat memakan waktu menit hingga jam kini mampu diselesaikan secara otomatis dengan rata-rata di bawah 15 detik. Kedua, dari sisi kemampuan deteksi, pendekatan verifikasi berbasis isi file terbukti mampu mendeteksi ancaman *zero-day* dan konten judi yang disamarkan di dalam gambar, yang tidak dapat ditangani oleh sistem berbasis *hash* maupun *signature*. Ketiga, dari sisi visibilitas, notifikasi *Telegram real-time* memastikan *administrator* selalu mendapatkan laporan insiden yang lengkap dan informatif meskipun tidak sedang memantau *dashboard*, sehingga mengurangi ketergantungan pada pemantauan manual.

4. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan maka dapat disimpulkan bahwa sistem keamanan berlapis berbasis *Wazuh* SIEM dengan fitur *Active Response* berhasil mengatasi dua kelemahan utama sistem terdahulu, yaitu sifat pasif yang hanya memberikan peringatan tanpa tindakan, serta keterbatasan deteksi berbasis *hash* yang tidak mampu mengenali ancaman judi online yang disamarkan. Sistem yang dibangun mampu mendeteksi seluruh skenario serangan dengan tingkat keberhasilan deteksi 100% pada Kondisi B maupun Kondisi C, sekaligus meremediasi ancaman secara otomatis dengan rata-rata waktu di bawah 15 detik pada Kondisi C. Integrasi notifikasi *Telegram Bot* memastikan *administrator* selalu mendapatkan laporan insiden secara *real-time* tanpa perlu memantau *dashboard* secara manual. sehingga keseluruhan sistem dapat bekerja secara otomatis dalam menghadapi berbagai variasi serangan konten judi *online* pada aplikasi web pemerintah desa.

Referensi

[1] P. Agustin and A. Wahyu, "Analisis Keamanan Situs '.go.id' Terhadap Serangan Web Defacement 'Judi Online,'" *InSearch (Informatic, Science, Entrepreneur, Applied Art, Research, Humanism)*, vol. 24, no. 1 Juli 2025, pp. 1–10, Jun. 2025, doi: 10.37278/insearch.v24i1.1125.

[2] A. Satari Rahim, P. Widodo, A. H. Reksoprodjo, and A. Warfare, "Identify Cyber Intelligence Threats in Indonesia," *International Journal Of Humanities Education And Social Sciences (IJHESS) E-ISSN*, vol. 3, no. 1, pp. 431–440, Aug. 2023, doi: doi.org/10.55227/ijhess.v3i1.426.

[3] M. Anis, A. Hilmi, P. Keamanan, F. Upload…, and R. K. Yunan, "Pengujian Keamanan Fitur Upload File Pada Sistem Aplikasi Web," *Jurnal Informatika: Jurnal pengembangan IT (JPIT)*, vol. 7, no. 1, Jan. 2022, doi: 10.30591/jpit.v7i1.3336.

[4] M. R. Reza Pahlevi, C. Umam, and L. B. Handoko, "Deteksi dan Pencegahan Web Defacing Judi Online dengan Wazuh SIEM dan Snort IDS Berbasis Signature," *Jurnal Algoritma*, vol. 22, no. 1, pp. 197–208, May 2025, doi: 10.33364/algoritma/v.22-1.2220.

[5] S. A. Hasbullah, M. N. Fauzan, and R. Andarsyah, "Implementation Layered Mitigation Techniques for Unrestricted File Upload and Server-Side JavaScript Injection," *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, vol. 5, no. 1, pp. 62–71, Jan. 2026, doi: 10.20885/snati.v5.i1.42248.

[6] C. Kurniawan and A. Triayudi, "File Integrity Monitoring as a Method for Detecting and Preventing Web Defacement Attacks," *Jurnal Online Informatika*, vol. 9, no. 2, pp. 276–285, Aug. 2024, doi: 10.15575/join.v9i2.1326.

[7] R. Aditya, Y. Muhyidin, and D. Singasatia, "Implementasi Security Information And Event Management (SIEM) Untuk Monitoring Keamanan Server Menggunakan Wazuh," *Jurnal Riset Sistem Informasi dan Teknik Informatika*, vol. 2, no. 5, 2024, doi: 10.61132/merkurius.v2i4.289.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

[8] M. Nas, F. Ulfiah, U. Putri, T. Elektro, P. Negeri, and U. Pandang, "Analisis Sistem Security Information and Event Management (SIEM) Aplikasi Wazuh pada Dinas Komunikasi Informatika Statistik dan Persandian Sulawesi Selatan," *Jurnal Teknologi Elekterika*, vol. 20, no. 2, 2023.

[9] M. Dehan Pratama, F. Nova, and D. Prayama, "Wazuh sebagai Log Event Management dan Deteksi Celah Keamanan pada Server dari Serangan Dos," 2022. [Online]. Available: http://jurnal-itsi.org

[10] B. Haryanto and D. W. Chandra, "Implementasi Wazuh Integritas File untuk Perlindungan Keamanan Berdasarkan Aktivitas Log di BTSI UKSW," *Jurnal Indonesia : Manajemen Informatika dan Komunikasi*, vol. 5, no. 1, pp. 183–192, Jan. 2024, doi: 10.35870/jimik.v5i1.447.

[11] L. Widyawati, Y. S. Prana, and M. Azwar, "Integration of Security Information and Event Management and WhatsApp Bot to Improve Server Security in Real-time," *RIGGS: Journal of Artificial Intelligence and Digital Business*, vol. 4, no. 2, pp. 97–111, May 2025, doi: 10.31004/riggs.v4i2.453.

[12] M. A. E. Putra, I. P. Hariyadi, and K. Marzuki, "OTOMATISASI KONFIGURASI WAZUH TERINTEGRASI VIRUSTOTAL MENGGUNAKAN ANSIBLE UNTUK MENDETEKSI DAN MEMPROTEKSI SERANGAN MALWARE," *Melek IT: Information Technology Journal*, vol. 11, no. 30 Juni 2025, pp. 55–66, Jun. 2025, doi: 10.30742/melekitjournal.v11i1.403.

[13] A. S. Maldini, W. S. J. Saputra, and D. A. Prasetya, "Multimodal Detection of Covert Online Gambling Advertisements Using Faster R-CNN and Tr-OCR," *bit-Tech*, vol. 8, no. 1, pp. 953–963, Aug. 2025, doi: 10.32877/bt.v8i1.2769.

[14] M. N. Arifin, M. Umar Mansyur, A. Rahman, N. P. Dewi, F. Prasetyo, and E. Putra, "Enhanced OCR Recognition for Madurese Text Documents: A Genetic Algorithm Approach with Tesseract 5.5," *JUITA: Jurnal Informatika*, vol. 13, no. 2, July 2025, pp. 109–118, Jul. 2025, doi: 10.30595/juita.v13i2.25794.

[15] H. A. Damanik and M. Anggraeni, "Sistem Deteksi Intrusi Hybrid dan Mitigasi Kerentanan Infrastruktur Jaringan Menggunakan Teknik Active Response (XDR) Wazuh dan Suricata," *Jurnal Pekommas*, vol. 9, no. 2, pp. 309–322, Dec. 2024, doi: 10.56873/jpkm.v9i2.5829.

[16] N. Ramadhanty, "Implementasi Kerangka Keamanan NIST Dan ISO/IEC 27001 Dalam Menghadapi Ancaman Risiko Siber," *Journal of Indonesian Management*, vol. 4, no. 4, Nov. 2024, doi: 10.53697/jim.v4i4.1973.

[17] A. N. Hasan and G. Purnama, "PERANCANGAN DAN SIMULASI JARINGAN INTERNET DENGAN MENERAPKAN METODE PENGEMBANGAN NDLC (NETWORK DEVELOPMENT LIFE CYCLE) PADA AKSES EDUCATION CENTRE," *JATI: Jurnal Mahasiswa Teknik Informatika*, vol. 8, no. 3, Juni 2024, pp. 2575–2585, Jun. 2024, doi: 10.36040/jati.v8i3.9488.

[18] D. D. Papaceda, A. Mawengkang, and S. Pratasik, "ANALISIS DAN PENGEMBANGAN JARINGAN KOMPUTER DI SMK NEGERI 8 WEDA HALMAHERA TENGAH," *EduTIK: Jurnal Pendidikan Teknologi Informasi dan Komunikasi*, vol. 3, pp. 1–13, 2023, doi: 10.53682/edutik.v3i1.6465.

[19] C. R. Prabowo, E. Rohadi, and D. Irmanto, "Study and Analysis of Docker Internal System Security From Docker Daemon Attack and DDOS on The Open Journal System," *JPUA: Jurnal Perpustakaan Universitas Airlangga: Media Informasi dan Komunikasi Kepustakawanan*, vol. 14, no. 1, pp. 61–68, Jun. 2024, doi: 10.20473/jpua.v14i1.2024.61-68.

[20] C. Prastika Bandhaso and Y. Sutanto, "Analisis Dan Implementasi Metode Random Early Detection (Red) Pada Jaringan TCP/IP," *Jutif: Jurnal Teknologi Informasi*, vol. 17, pp. 51–57, Jun. 2022, doi: 10.35842/jtir.v17i2.458.

[21] N. A. Sultan, N. A. Sultan, and R. Putros Qasha, "CONTAINER-BASED VIRTUALIZATION FOR BLOCKCHAIN TECHNOLOGY: A SURVEY," *Jordanian Journal of Computers and Information Technology (JJCIT)*, vol. 09, no. 03, September 2023, pp. 234–247, Sep. 2023, doi: 10.5455/jjcit.71-1682778947.

[22] A. Shafiyyah, G. F. Nama, and R. A. Pradipta, "IMPLEMENTASI WAZUH MENGGUNAKAN METODE PPDIOO DI SISTEM KEAMANAN JARINGAN PSDKU UNIVERSITAS LAMPUNG WAYKANAN SEBAGAI DETEKSI DAN RESPON SERANGAN SIBER," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 2, Apr. 2024, doi: 10.23960/jitet.v12i2.4074.

DOI: https://doi.org/10.31004/riggs.v5i2.12023
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)