pISSN: 2442-3386 eISSN: 2442-4293
Melek IT
Vol 11 No 1 Jan – Jun 2025, 31 - 42
INFORMATION TECHNOLOGY JOURNAL

IMPLEMENTASI SECURITY INFORMATION AND EVENT MANAGEMENT UNTUK MENCEGAH SERANGAN DEFACE PADA SERVER TERINTEGRASI TELEGRAM

Meidi Djamalyanto*[1], Lilik Widyawati[2], Husain[3] I Putu Hariyadi[4]
[1]Program Studi Ilmu Komputer, Fakultas Teknik, Universitas Bumigora, yantomeidi22@gmail.com
[2]Program Studi Ilmu Komputer, Fakultas Teknik, Universitas Bumigora, lilikwidya@universitasbumigora.ac.id
[3]Program Studi Ilmu Informatika, Fakultas Teknik, Universitas Bumigora, husain@universitasbumigora.ac.id
[4]Program Studi Ilmu Komputer, Fakultas Teknik, Universitas Bumigora,
Iputuhariyadi@universitasbumigora.ac.id
*)Korespondensi: yantomeidi22@gmail.com

Abstrak

Serangan deface merupakan ancaman kritis yang mengganggu integritas server dan merusak reputasi organisasi. Penelitian ini bertujuan merancang sistem Security Information and Event Management (SIEM) berbasis Wazuh yang terintegrasi dengan Telegram Bot untuk mendeteksi, mencegah, dan memberikan notifikasi real-time terhadap serangan deface. Metode yang digunakan adalah Network Development Life Cycle (NDLC) dengan tiga tahap utama: analisis kebutuhan, desain sistem, dan simulasi prototipe. Implementasi dilakukan pada lingkungan virtual menggunakan Ubuntu Server 22.04 sebagai Wazuh Manager dan Parrot Security OS sebagai simulator serangan. Hasil penelitian menunjukkan bahwa Wazuh berhasil mendeteksi tiga jenis serangan utama: File Upload Vulnerability, Remote Code Execution (RCE), dan Webshell melalui analisis log dan aturan kustom. Integrasi dengan Telegram Bot memungkinkan pengiriman notifikasi instan saat ancaman terdeteksi, disertai respons otomatis seperti pemulihan direktori (restore), pemblokiran alamat IP penyerang, dan mitigasi proaktif. Uji coba sebelum penerapan Wazuh membuktikan kerentanan server terhadap modifikasi file, sementara setelah implementasi, sistem mampu mencegah perubahan ilegal dengan efektivitas 100%. Kesimpulan penelitian ini menegaskan bahwa kombinasi SIEM Wazuh dan Telegram Bot meningkatkan keamanan server melalui deteksi dini, respons cepat, dan pemantauan terpusat. Solusi ini tidak hanya mengurangi risiko deface tetapi juga menyediakan mekanisme notifikasi yang efisien bagi administrator. Rekomendasi untuk pengembangan meliputi penambahan variasi serangan, peningkatan active-response, dan optimasi integrasi dengan platform lain.
Kata Kunci: SIEM, Wazuh, Deface, Telegram Bot, Keamanan Server.

Abstract

Deface attacks are critical threats that disrupt server integrity and damage an organization's reputation. This research aims to design a Wazuh-based Security Information and Event Management (SIEM) system integrated with the Telegram Bot to detect, prevent, and provide real-time notifications against deface attacks. The method used is the Network Development Life Cycle (NDLC), which has three main stages: requirement analysis, system design, and prototype simulation. The implementation was conducted on a virtual environment using Ubuntu Server 22.04 as the Wazuh Manager and Parrot Security OS as the attack simulator. The results showed that Wazuh successfully detected three main types of attacks: File Upload Vulnerability, Remote Code Execution (RCE), and Webshell through log analysis and custom rules. Integration with Telegram Bot enables instant notification when threats are detected, along with automated responses such as directory restores, attacker IP address blocking, and proactive mitigation. Tests prior to Wazuh's implementation proved the server's vulnerability to file modification, while after implementation, the system was able to prevent illegal changes with 100% effectiveness. The conclusion of this study confirms that the combination of Wazuh SIEM and Telegram Bot improves server security through early detection, rapid response, and centralized monitoring. This solution not only reduces the risk of deface but also provides an efficient notification mechanism for administrators. Recommendations for development include adding attack variations, improving active response, and optimizing integration with other platforms.
Keywords: *SIEM, Wazuh, Deface, Telegram Bot, Server Security..*

I. PENDAHULUAN

Di era digital yang berkembang pesat, internet telah menjadi pusat aktivitas yang sangat penting bagi organisasi dan individu. Server yang mendukung aplikasi web memegang peranan vital dalam menyimpan dan mengelola data penting, seperti konten situs web, informasi pengguna, dan konfigurasi system [1]. Dengan semakin besarnya ketergantungan pada teknologi web, ancaman terhadap keamanan server juga meningkat, memerlukan solusi perlindungan yang lebih maju untuk menjaga integritas dan keamanan data [2]. Dalam konteks ini, peran Sistem Manajemen Informasi dan Keamanan (SIEM) menjadi sangat penting, karena sistem ini dapat mendeteksi dan menangani berbagai ancaman, termasuk serangan deface [3].

Serangan deface merujuk pada aktivitas yang dilakukan oleh peretas dengan tujuan merusak atau memodifikasi tampilan halaman utama suatu situs web. Dalam serangan ini, penyerang berhasil mengakses server dan mengganti tampilan halaman web dengan konten yang tidak sah atau merusak[4]. Pada kerentanan dalam aplikasi web sering kali menjadi pintu masuk untuk serangan ini, seperti kasus Sekretariat Kabinet RI pada Juli 2021, situs Setkab.go.id milik Sekretariat Kabinet Republik Indonesia mengalami serangan deface yang menyebabkan perubahan tampilan halaman web. Tampilan situs berubah menjadi hitam dengan gambar demonstran dan tulisan "Padang Blackhat ll Anon Illusion Team Pwned By Zyy Ft Luthfifake." Polisi menduga serangan ini bertujuan untuk keuntungan ekonomi, seperti menjual script backdoor. Penyelidikan menunjukkan bahwa peretasan terjadi akibat kelemahan sistem keamanan dan kelalaian operator. Situs Setkab telah diperbaiki dan tidak ada dokumen rahasia yang terpengaruh. Pada Oktober 2021, situs Pusat Malware Nasional (Pusmanas) milik Badan Siber dan Sandi Negara (BSSN) menjadi korban serangan deface. Hacker merubah tampilan situs dengan pesan "Hacked by theMx0nday" dan menyatakan bahwa serangan ini sebagai balasan untuk peretasan situs Brasil. Menghadapi ancaman ini memerlukan solusi yang efektif untuk memantau, mendeteksi, dan merespons potensi serangan secara proaktif [5]. Di sinilah SIEM memainkan peran penting dengan kemampuannya untuk mendeteksi dan merespons aktivitas mencurigakan secara real-time[6].

Security Information and Event Management (SIEM) adalah teknologi yang dirancang untuk mengelola risiko keamanan dengan mengumpulkan, menganalisis, dan mengelola data log dari berbagai sumber dalam satu platform terintegrasi[7]. Wazuh, sebagai salah satu platform SIEM open source, menawarkan kemampuan untuk pemantauan real-time, analisis data log, dan deteksi ancaman yang dapat membantu melindungi server dari serangan deface [8]. SIEM secara umum membantu mengatasi ancaman seperti serangan deface dengan mengidentifikasi aktivitas mencurigakan yang menunjukkan tanda-tanda upaya defacement, memberikan peringatan dini kepada tim keamanan, dan memungkinkan mereka untuk mengambil tindakan preventif serta korektif dengan cepat[9]. Wazuh memungkinkan administrator server untuk mengidentifikasi pola aktivitas yang mencurigakan, mendeteksi potensi serangan lebih awal, dan memberikan respons yang cepat untuk mengatasi masalah tersebut [10].

Untuk lebih meningkatkan efektivitas sistem SIEM dalam menangani serangan deface, penelitian ini juga akan membahas integrasi Wazuh dengan telegram bot[11]. Integrasi Telegram Bot memungkinkan administrator menerima notifikasi realtime langsung di aplikasi telegram-platform komunikasi popular dengan akses yang mudah[12].

Penggunaan telegram bot dalam konteks ini memungkinkan administrator untuk menerima peringatan tentang aktivitas mencurigakan, hasil analis log dan status keamanan server tanpa perlu terus-menerus memantau dashboard Wazuh secara manual[13].Notifikasi yang dikirim melalui Telegram dapat mencakup detail tentang potensi serangan, seperti upaya pegunggahan file berbahaya atau penyuntikan skrip jahat, dan memungkinkan administrator untuk mengambil tindakan cepat langsung dari aplikasi Telegram.

Integrasi ini juga akan memungkinkan administrator untuk menjalankan perintah atau merespons insiden langsung dari Telegram, meningkatkan efisiensi dalam pengelolaan keamanan[14]. Misalnya, jika sistem mendeteksi serangan deface, atau aktifivitas mencurigakan, telegram bot dapat mengirimkan nontifikasi kepada administrator yang kemudian dapat merespons dengan cepat untuk menanggapi ancaman[15].

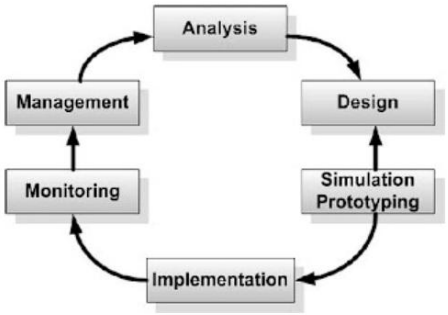
Dengan penerapan Wazuh yang diintegrasikan dengan Telegram bot, penelitian ini bertujuan untuk memperkuat perlindungan terhadap server dari serangan deface, mempermudah proses monitoring dan response, serta meningkatkan efektivitas dalam pengelolaan kemananan informasi. Diharapkan, integrasi ini tidak hanya membantu dalam deteksi dan mitigasi ancaman secara efisien, tetapi juga memberikan kontribusi signifikan dalam menjaga integritas dan kepercayaan terhadap aplikasi web yang dikelola oleh organisasi / instansi.

II. METODE

Metode penelitian yang diterapkan dalam penelitian ini adalah *Network Development Life*

pISSN: 2442-3386 eISSN: 2442-4293
Vol 11 No 1 Jan – Jun 2025, 33 - 42
Melek IT
INFORMATION TECHNOLOGY JOURNAL

Cycle (NDLC). NDLC adalah pendekatan terstruktur dalam pengembangan sistem yang mencakup serangkaian tahapan mulai dari analisis hingga manajemen, dengan fokus pada pengembangan dan implementasi solusi teknologi.



Gambar 1 Tahapan Network Development Life Cycle yang digunakan dalam penelitian, yaitu Analisis, Desain, dan Prototyping.

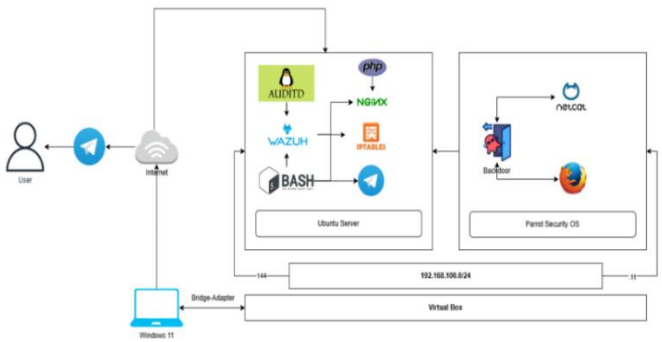
Penulis hanya menggunakan 3 (tiga) dari 6 (enam) tahapan pertama, yaitu:

1. *Analysis*
   Pada tahap analisis, data terkait teknologi terkini dalam manajemen dan pemantauan server melawan serangan deface di kumpulkan melalui studi literlatur. Penelitian ini melibatkan pengumpulan dan kajian artikel ilmiah, buku, paper, serta sumber informasi lain yang relevan untuk memahami konsep dan teknologi *Security Information and Event Management* (SIEM), khususnya *Wazuh*, dalam konteks pencegahan serangan *deface*, Selain itu, analisis juga mencakup studi mengenai integrasi *Wazuh* dengan *Telegram bot* sebagai metode untuk meningkatkan notifikasi dan respons terhadap ancaman secara *real-time.*
2. *Design*

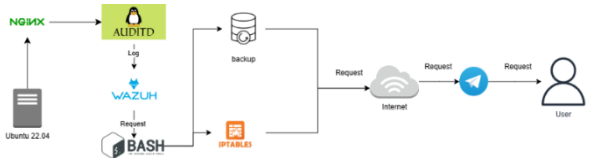
Tahap desain melibatkan perancangan sistem pemantauan keamanan server yang terintegrasi dengan *Telegram bot* berdasarkan hasil analisis data. Desain sistem mencakup pembuatan skema integrasi antara *Wazuh* dan *Telegram bot*, serta perancangan alur notifikasi dan pengelolaan insiden. Rancangan ini juga mempertimbangkan pengalokasian sumber daya jaringan, konfigurasi IP, serta kebutuhan perangkat keras *(hardware)* dan perangkat lunak *(software)* yang diperlukan untuk implementasi sistem. Fokus utama adalah merancang sistem yang dapat memantau, mendeteksi, dan memberikan respons terhadap serangan deface dengan efektif melalui notifikasi yang diterima di *Telegram bot*.

A. Rancangan Ujicoba

Pada Gambar 2 terlihat rancangan jaringan uji coba menggunakan 1 laptop yang sebagai host utama. pada laptop tersebut, diinstal *Virtualbox* sebagai platform virtualisasi untuk menjalankan beberapa mesin virtual yang terhubung melalui *Bridge-Adapter*. *Bridge Adapter* berfungsi untuk menguhubungkan VM ke jaringan Internet.Terdapat 2 mesin virtual, yaitu *Ubuntu Server* dengan alamat IP 192.168.100.144/24 sebagai *Web Server* dan Wazuh Manager yang menampung berbagai layanan Web Server, *Parrot Security OS* dengan alamat 192.168.100.11/24 berperan untuk melakukan penyerangan/pengujian. W*azuh Server* juga memantau aktifitas yang mencurigakan seperti serangan *File Vulnerability*, RCE, akses ilegal dan *Ubuntu Server* juga menjalankan program *Telegram Bot* API yang digunakan untuk mengirim notifikasi *log* secara otomatis ke dalam *Telegram* sekaligus melakukan otomatisasi jika pengguna mengirimkan perintah sesuai yang ditentukan.



Gambar 2 Topologi jaringan uji coba menggunakan VirtualBox dengan dua mesin virtual (Ubuntu Server dan Parrot OS) yang terhubung melalui *Bridge-Adapter*.



Gambar 3 Alur integrasi antara pengguna, *Bash script,Auditd,Iptables, dan Telegram Bot* dalam sistem kemanan berbasis Wazuh.

B. Rancangan Sistem Integrasi *Bot*

Pada Gambar 3 User mengirimkan permintaan (*request*) melalui Internet ke *BASH Script* yang berjalan di *Ubuntu* 22.04. Permintaan tersebut diproses dengan bantuan *AUDITD* untuk memantau aktivitas sistem dan *IPTABLES* untuk mengubah aturan *firewall*. Hasilnya, *alert* atau *log* keamanan disimpan dan dapat dikirim kembali ke pengguna melalui integrasi dengan *Telegram Bot*.Selain itu, pengguna juga dapat mengirim perintah otomatisasi melalui *BASH Script* untuk menambah/menghapus konfigurasi *active-response*,sementara Backup memastikan konfigurasi tersimpan dengan aman.

Telegram Bot pada sistem ini dapat diakses sebagai antarmuka komunikasi antara sistem pemantauan (Wazuh) dan administrator server. Telegram Bot menerima input berupa log atau peringatan keamanan dari Wazuh melalui API Telegram yang telah dikonfigurasi menggunakan token dari BotFather. Sistem memanfaatkan skrip otomatis berbasis Bash (telegram-alert.sh) yang dijalankan oleh fitur active-response Wazuh saat terdeteksi adanya aktivitas mencurigakan.

Begitu sistem mendeteksi serangan seperti File Upload Vulnerability, Remote Code Execution, atau Webshell, Wazuh akan menjalankan skrip yang mengirimkan pesan dalam format JSON ke Telegram. Pesan tersebut memuat informasi rinci seperti waktu kejadian, alamat IP penyerang, dan jenis serangan yang teridentifikasi.

Dari sisi operasional, Telegram Bot memberikan kemampuan kepada administrator untuk merespons ancaman secara langsung melalui perintah yang dikirim via aplikasi Telegram. Contohnya, administrator dapat mengetik perintah /blockip 192.168.100.11 untuk memblokir IP penyerang atau perintah /restore untuk memulihkan file yang terpengaruh. Seluruh proses ini difasilitasi oleh integrasi antara skrip Bash, pengaturan active-response di file ossec.conf, dan koneksi webhook Telegram.

Penggunaan Telegram memberikan keuntungan signifikan dalam bentuk mobilitas, pemberitahuan waktu nyata (real-time), dan kemudahan penggunaan tanpa harus mengakses dashboard Wazuh secara manual. Hal ini sangat mendukung operasional sistem yang membutuhkan tanggapan cepat terhadap ancaman keamanan.

C. Rancangan Pengalamatan IP

Dalam rancangan implementasi sistem pemantauan serangan server yang terintegrasi dengan *Telegram Bot*, alokasi alamat IP menggunakan dua *subnet* Kelas C yaitu 192.168.100.144/24 dan 192.168.100.83/24. Konfigurasi lengkap distribusi alamat IP untuk setiap komponen infrastruktur dapat dilihat pada Tabel 1.

Tabel 1 Distribusi Alamat IP untuk Komponen Infrastruktur Server dan Simulator Serangan

Nama Perangkat	Alamat IP	*Subnet mask*
wazuh-Server	192.168.100.144	255.255.255.0
Parrot Sec OS	192.168.100.11	255.255.255.0

3. *Simulation Prototyping*

Pada tahap ini terdiri dari beberapa bagian di antaranya yaitu instalasi dan konfigurasi pada masing-masing perangkat pendukung, uji coba dan analisa uji coba. Uji coba terdiri dari dua bagian yaitu uji coba sebelum dan sesudah menambahkan active-response pada *Wazuh Server*. Selain itu, di lakukan konfigurasi dan pembuatan kode pada server untuk mengintegrasikan *Telegram Bot. Telegram Bot* di integrasikan melalui program Python lalu di hubungkan ke dalam *Telegram* API yang dibuat menggunakan bahasa *Javascript*.

A. Tahap Instalasi dan Konfigurasi

Tahap instalasi dan konfigurasi hanya satu bagian, yaitu instalasi dan konfigurasi *Host Wazuh Server*. Konfigurasi pada Host *Wazuh Server* melibatkan pengaturan alamat IP pada antarmuka *enp0s3* dengan mengonfigurasi alamat IP menjadi statis. Proses instalasi dan konfigurasi *Wazuh Server* mencakup penyesuaian berkas

pISSN: 2442-3386 eISSN: 2442-4293
Vol 11 No 1 Jan – Jun 2025, 35 - 42
Melek IT
INFORMATION TECHNOLOGY JOURNAL

ossec.config serta integrasi *Telegram Bot* sebagai media pemantauan (monitoring) pada platform *Telegram*

B. Skenario Uji Coba

Pada tahap ini, akan dilakukan pengujian terhadap sistem *Telegram Bot* dan pemantauan dengan beberapa skenario sebagai berikut: Penambahan atau penghapusan konfigurasi Active Response melalui otomatisasi *Telegram Bot* berdasarkan data yang dikirimkan. Pemantauan serangan *File Vulnerability*, *Remote Code Execution*, dan *Web Shell* yang akan dikirimkan melalui Telegram Bot secara real-time. Pengiriman data hasil *log* dari *Wazuh Server* untuk menampilkan perbandingan hasil uji coba serangan sebelum dan sesudah penerapan konfigurasi *Active Response*. Hal ini bertujuan menampilkan perbedaan dampak serangan pada kedua kondisi tersebut.

III. HASIL DAN PEMBAHASAN

3.1. Hasil Instalasi *Packages* pada *Server*

Server perlu di lakukan instalasi & konfigurasi mengunakan beberapa packages terlebih dahulu agar dapat melakukan uji coba penyerangan. Beberapa *packages* yang diinstall digunakan sebagai pendukung dalam pengujian system pada server nantinya.

```
yantok@yantok:~$ mysql --version
mysql  Ver 15.1 Distrib 10.6.18-MariaDB, for debian-linux-gnu (x86_64) using
```

Gambar 4 Tampilan hasil MySQL sebagai basis data pendukung pada server.

Pada Gambar 4 merupakan hasil instalasi dari *mysql* yang di gunakan sebagai *database* untuk menampung data-data dari program.

```
yantok@yantok:~$ php -v
PHP 8.1.2-1ubuntu2.20 (cli) (built: Dec  3 2024 20:14:35) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.1.2, Copyright (c) Zend Technologies
    with Zend OPcache v8.1.2-1ubuntu2.20, Copyright (c), by Zend Technologies
```

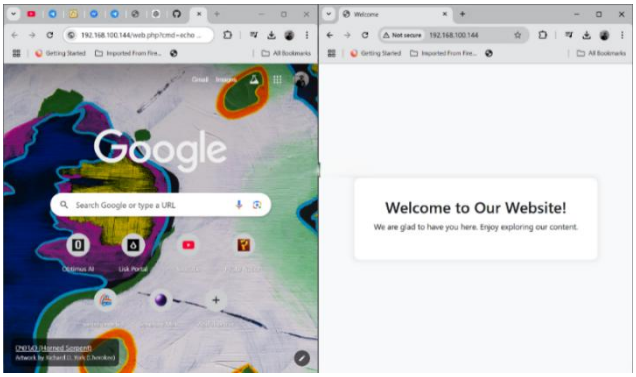
Gambar 5 Hasil instalasi *PHP* yang digunakan untuk menjalankan skrip dalam server web Nginx

Pada Gambar 5 merupakan hasil instalasi dari PHP yang digunakan untuk membaca script dari program *Nginx* yang sudah disediakan dihalaman web

```
yantok@yantok:~$ sudo apt install nginx php php-fpm
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
php is already the newest version (2:8.1+92ubuntu1).
php-fpm is already the newest version (2:8.1+92ubuntu1).
nginx is already the newest version (1.18.0-6ubuntu14.5).
0 upgraded, 0 newly installed, 0 to remove and 104 not u
```

Gambar 6 Hasil instalasi *Nginx* yang berfungsi sebagai web server dalam pengujian serangan.

Pada Gambar 6 merupakan hasil instalasi *Nginx* yang di gunakan sebagai *web server* pada *server Wazuh* sehingga dapat menampung beberapa *file* web yang akan di gunakan untuk uji coba penyerangan.



Gambar 7 Tampilan halama utama *Nginx* setelah instalasi, sebagai validasi bahwa *server* aktif.

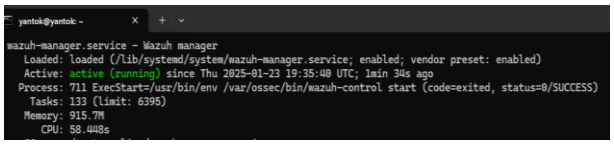
Pada Gambar 7 merupakan hasil dari program *Nginx* yang sudah di lakukan. Terlihat halaman *Nginx* berhasil di akses dan dapat di gunakan sebagai alat untuk melakukan uji coba serangan *File Vulnerability* dan *Remote Code Execution.*

3.2. Hasil Instalasi *Wazuh Manager*

```
yantok@yantok:~$ sudo bash ./wazuh-install.sh -a
16/01/2025 20:14:05 INFO: Starting Wazuh installation assistant. Wazuh version: 4.7.5
16/01/2025 20:14:05 INFO: Verbose logging redirected to /var/log/wazuh-install.log
16/01/2025 20:14:18 INFO: Wazuh web interface port will be 443.
16/01/2025 20:14:29 INFO: Wazuh repository added.
16/01/2025 20:14:29 INFO: --- Configuration files ---
16/01/2025 20:14:29 INFO: Generating configuration files.
16/01/2025 20:14:32 INFO: Created wazuh-install-files.tar. It contains the Wazuh cluster key,
tion.
16/01/2025 20:14:32 INFO: --- Wazuh indexer ---
16/01/2025 20:14:33 INFO: Starting Wazuh indexer installation.
16/01/2025 20:16:09 INFO: Wazuh indexer installation finished.
16/01/2025 20:16:09 INFO: Wazuh indexer post-install configuration finished.
16/01/2025 20:16:09 INFO: Starting service wazuh-indexer.
16/01/2025 20:16:31 INFO: wazuh-indexer service started.
16/01/2025 20:16:31 INFO: Initializing Wazuh indexer cluster security settings.
16/01/2025 20:16:42 INFO: Wazuh indexer cluster initialized.
16/01/2025 20:16:42 INFO: --- Wazuh server ---
16/01/2025 20:16:42 INFO: Starting the Wazuh manager installation.
16/01/2025 20:19:23 INFO: Wazuh manager installation finished.
16/01/2025 20:19:23 INFO: Starting service wazuh-manager.
16/01/2025 20:19:45 INFO: wazuh-manager service started.
16/01/2025 20:19:45 INFO: Starting Filebeat installation.
16/01/2025 20:19:59 INFO: Filebeat installation finished.
16/01/2025 20:20:01 INFO: Filebeat post-install configuration finished.
16/01/2025 20:20:01 INFO: Starting service filebeat.
16/01/2025 20:20:06 INFO: filebeat service started.
16/01/2025 20:20:06 INFO: --- Wazuh dashboard ---
16/01/2025 20:20:06 INFO: Starting Wazuh dashboard installation.
16/01/2025 20:23:22 INFO: Wazuh dashboard installation finished.
16/01/2025 20:23:22 INFO: Wazuh dashboard post-install configuration finished.
16/01/2025 20:23:22 INFO: Starting service wazuh-dashboard.
16/01/2025 20:23:24 INFO: wazuh-dashboard service started.
16/01/2025 20:24:00 INFO: Initializing Wazuh dashboard web application.
16/01/2025 20:24:02 INFO: Wazuh dashboard web application initialized.
16/01/2025 20:24:02 INFO: --- Summary ---
16/01/2025 20:24:02 INFO: You can access the web interface https://<wazuh-dashboard-ip>:443
    User: admin
    Password: gNeX6?jogTfJML0TnFi0uR8t?dwQqhZG
16/01/2025 20:24:02 INFO: Installation finished.
```

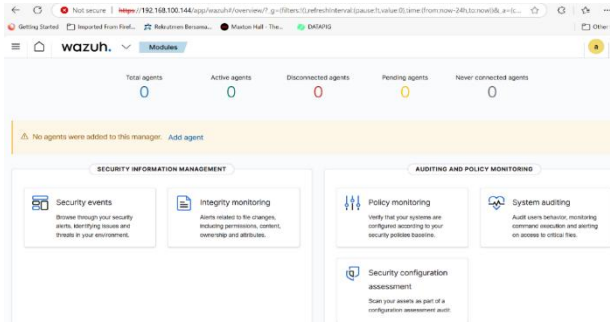
Gambar 8 Output instalasi Wazuh Manager yang menampilkan username dan password login ke dashboard.

Pada Gambar 8 *Wazuh manager* sudah berhasil di pasang dengan menggunakan perintah untuk menginstall semua layanan pada *Wazuh Server* dan menampilkan *user* dan*password* dari hasil instalasi *Wazuh Server*. *User* dan *Password* di gunakan sebagai identitas agar bisa masuk ke dalam halaman *Wazuh Server* melalui *Wazuh Dashboard.*



Gambar 9 *Status* layanan Wazuh Manager yang sudah aktif dan berjalan di server.

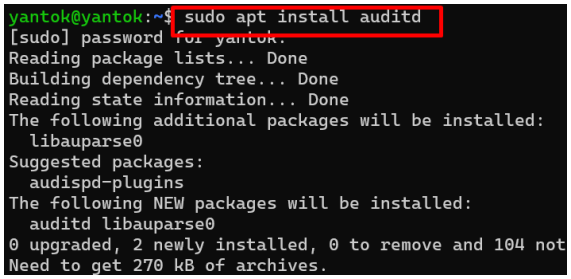
Terlihat pada Gambar 9 status dari *Wazuh Manager* telah aktif *(running)* sehingga dapat di simpulkan bahwa *Wazuh Manager* Telah berjalan Pada Server.



Gambar 10 *Dashboard* utama Wazuh untuk memonitoring kemanan server secara *real-time.*

Terlihat pada Gambar 10 merupakan tampilan dari halaman *Wazuh Dashboard* yang merupakan halaman utama dari seluruh manajemen yang sudah di sediakan oleh *Wazuh Manager.* Halaman tersebut di gunakan untuk menjalankan semua fitur yang ada pada *Wazuh Manager*.

3.3. Instalasi Package dan Konfigurasi Wazuh Server



Gambar 11 Insatalasi paket Auditd untuk memantau aktivitas sistem pada server.

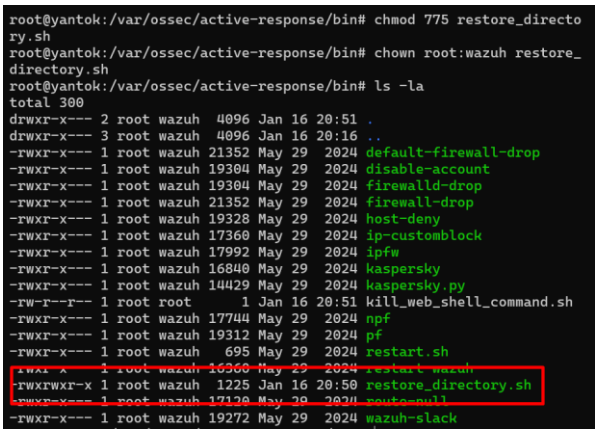
Pada Gambar 11 terlihat server melakukan instalasi terhadap *package auditd* yang berfungsi sebagai alat pemantau keamanan untuk melacak dan mencatat aktivitas sistem secara rinci sehingga dapat di deteksi kejanggalan yang terjadi pada *directory web server.*

3.4. Konfigurasi dan Penerapan Active Response

Penerapan *active response* di perlukan agar dapat mengirimkan *response* ketika serangan terdeteksi sehingga serangan dapat di cegah. Penerapan ini perlu di lakukan pada konfigurasi wazuh server dengan menggunakan perintah-perintah khusus dan berdasarkan tipe serangan yang akan di pilih. Adapun *active response* yang di buat adalah sebagai berikut :

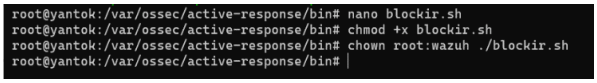
3.4.1 Restore Directory Response

Pada *Restore Directory Response* digunakan untuk mengembalikan *file web ke versi aman* yang sudah di rubah oleh penyerang secara otomatis sehingga tidak ada terjadinya perubahan pada halaman atau konten website yang terkena serangan Deface



Gambar 12 Pembuatan skrip *Bash restore_directory.sh* untuk dijalankan oleh *Wazuh Server.*

Pada Gambar 12 merupakan langkah pembuatan dari file active response restore_directory.sh dengan mengkonfigurasi mode dan owner dari file *restore_directory.sh* sehingga dapat di jalankan oleh *user wazuh.*



Gambar 13 Pembuatan file blokir.sh sebagai skrip untuk memblokir alamat IP penyerang.

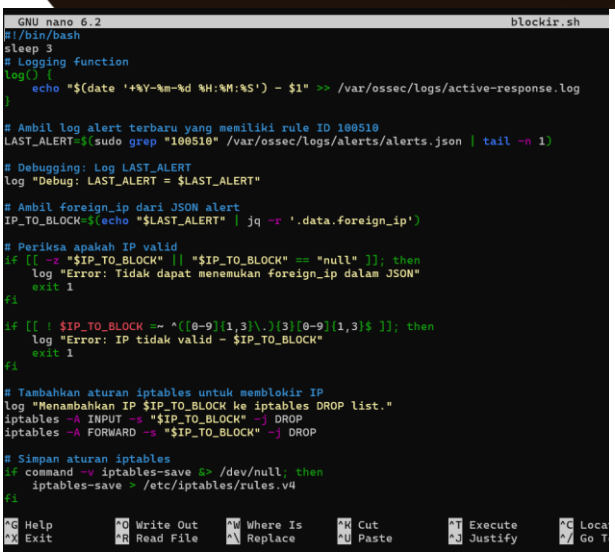
Pada Gambar 13 *Blockir Response* digunakan untuk memblockir alamat dari penyerang yang biasa digunakan untuk melakukan *deface* menggunakan webshell atau koneksi contohnya *netcat.* Dengan melakukan pemblockiran maka penyerang tidak dapat terhubung lagi pada server untuk mengubah atau menambahkan file sensitif.

Terlihat pada Gambar 14 merupakan sedikit hasil konfigurasi dari file blockir.sh dengan menggunakan bahasa *bash* yang akan di jalankan oleh *wazuh server* nanti sesuai fungsinya.

3.4.2 Telegram Alert Response

Telegram Alert Response digunakan untuk mengirimkan pesan-pesan kepada user melalui *telegram* dengan API yang sudah di dapatkan melalui *bot father*. Dengan begitu *user* akan selalu mendapatkan pesan secara *real-time* ketika terjadi serangan khusus.

pISSN: 2442-3386 eISSN: 2442-4293
Vol 11 No 1 Jan – Jun 2025, 37 - 42
Melek IT
INFORMATION TECHNOLOGY JOURNAL



Gambar 14 Isi konfigurasi dari skrip blokir.sh dengan perintah bash.

Gambar 15 Pembuatan skrip telegram.sh untuk mengirim notifikasi ke telegram.

Pada Gambar 15 merupakan langkah pembuatan dari *file active response telegram-alert.sh* dengan mengkonfigurasi *mode* dan *owner* dari *file telegram-alert.sh* sehingga dapat di jalankan oleh *user wazuh*.

Gambar 16 Kode konfigurasi dari *telegram-alert.sh* sebagai bagian dari *active-response*

Terlihat pada Gambar 16 merupakan sedikit hasil konfigurasi dari *file telegram-alert.sh* dengan menggunakan bahasa bash yang akan di jalankan oleh *wazuh server* nanti sesuai fungsinya.

3.4.3 Penerapan Active-Response

Penerapan *active-response* perlu di lakukan pada konfigurasi *file ossec.conf* agar dapat menjalankan perintah-perintah yang di inginkan sesuai dengan id dari rule yang sudah di dapatkan pada laporan serangan yang terkumpul di *wazuh server*. *Active-response* inilah yang berkontribusi untuk melakukan pencegahan terhadap serangan yang di arahkan kepada server sehingga *wazuh* tidak hanya berfungsi sebagai pemantau saja, akan tetapi juga berhungsi sebagai penangkal dari serangan tersebut.

```
<active-response>
    <command>telegram-alert</command>
    <location>local</location>
    <rules_id>100500,100501,100502,100501,100502</rules_id>
    <timeout>10</timeout>
</active-response>

<active-response>
    <command>restore-directory</command>
    <location>local</location>
    <rules_id>100500,100501,100502,100501,100502</rules_id>
    <timeout>10</timeout>
</active-response>


<active-response>
    <command>blockir</command>
    <location>local</location>
    <rules_id>100510,100521</rules_id>
    <timeout>10</timeout>
</active-response>
```

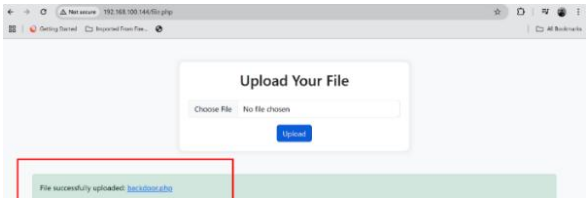
Gambar 17 Konfigurasi *file ossec* yang menunjukkan implementasi *active-response*.

Pada Gambar 17 merupakan hasil konfigurasi yang di lakukan pada *file ossec.conf* untuk menerapkan *active-response* berdasarkan *rule id* yang sudah terdeteksi pada laporan *wazuh* servernya sehingga dapat membedakan antara serangan-serangan yang ada untuk di lakukan pencegahan menggunakan command yang sudah di identifikasikan sebelum. Pada gambar 23 juga berfokus pada beberapa serangan dan 3 command yaitu blockir, restore-directory dan telegram-alert yang digunakan sesuai dengan fungsinya masing-masing..

3.5. Uji Coba Serangan Sesudah Penerapan Wazuh

A. *File Vulnerability*

Pada serangan *File Vulnerability*, penyerang akan menggunakan celah keamanan yang ada pada halaman */file.php* pada website menggunakan browser dari penyerang.

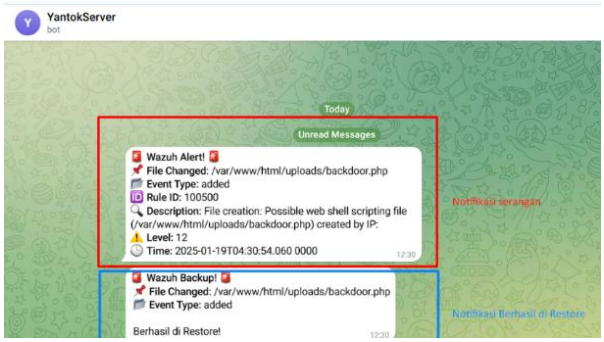


Gambar 18 Simulasi serangan *File Upload Vulnerability* melalui halaman */file.php*

Terlihat pada Gambar 18 penyerang berhasil mengupload *file backdoor.php* yang di mana file tersebut adalah file yang berbahaya karena dapat melakukan modifikasi, menambah dan mengakses file pada server sehingga dapat mengakibatkan serangan *Deface* jika mengubah konten dari website.

Gambar 19 Response sistem terhadap file backdoor yang sudah dihapus secara otomatis.

Pada Gambar 19 menunjukkan hasil dari file backdoor ketika di akses setelah proses upload selesai menunjukkan tidak ada file yang memiliki nama serupa sehingga dapat di simpulkan bahwa file tersebut telah terhapus sebelum di akses oleh penyerang.



Gambar 20 Notifikasi serangan dari Telegram Bot yang menunjukan aktivitas upload file berbahaya

Pada Gambar 20 merupakan hasil screenshot dari *telegram* yang menunjukkan bahwa notifikasi dari serangan yang di kirimkan ke server. Terlihat juga notifikasi menunjukkan hasil *response* yang telah berhasil di lakukan.

B. Remote Code Execution

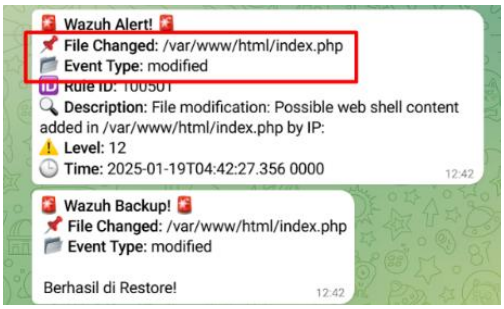
Pada serangan *Remote Code Execution* (RCE) penyerang melakukan serangan pada halaman */web.php* yang dimana file tersebut memiliki celah keamanan RCE yang bisa menjalankan perintah-perintah dari *linux*. Penyerang akan melakukan modifikasi terhadap *file index.php* sehingga akan mengubah konten dari *file index* webnya.

Gambar 21 Eksekusi *Remote Code Execution* melalui URL manipulasi untuk membuat *file hacker.php*

Pada Gambar 21 Menunjukkan penyerang melakukan serangan dengan memanfaatkan perintah linux untuk membuat sebuah file baru dengan nama *hacker.php* dan dengan konten yang di masukkan pada URL. Terlihat juga ketika menampilkan *file* pada *directory* bahwa *file hacker.php* berhasil di buat.

Gambar 22 Response sistem file hacker.php gagal diakses karena telah dihapus otomatis.

Terlihat pada Gambar 22 merupakan hasil ketika mengakses *file hacker.php* yang di buat sebelumnya. Hasil yang di tampilkan adalah file yang di akses tidak di temukan sehingga dapat kita simpulkan bahwa file tersebut sudah berhasil di hapus sebelum di akses oleh penyerang.

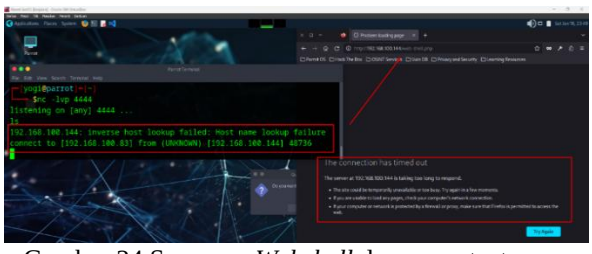


Gambar 23 Notifikasi Telegram atas serangan RCE dan laporan backup data otomatis.

Pada Gambar 23 merupakan hasil notifikasi yang di kirimkan melalui telegram bot ke pada user berdasarkan laporan dari serangan *Remote Code Execution* (RCE) pada *wazuh*.

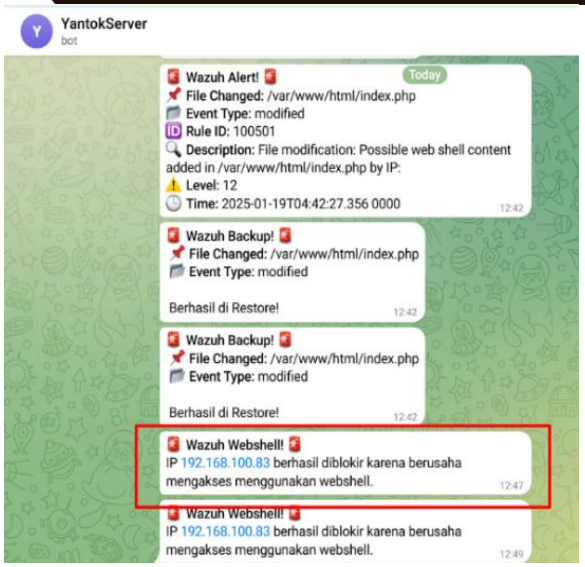
C. *Webshell Netcat*

Pada serangan *Webshell Netcat* penyerang memanfaatkan *file* yang sudah ada dan berbahaya yang dimana *file* tersebut dapat menghubungkan antara *server* dan juga penyerang. Serangan ini di lakukan menggunakan *netcat* untuk menghubungkan antara *server* dan *client.*



Gambar 24 Serangan *Webshell* dengan netcat yang berhasil digagalkan oleh sistem.

pISSN: 2442-3386 eISSN: 2442-4293
Vol 11 No 1 Jan – Jun 2025, 39 - 42
Melek IT
INFORMATION TECHNOLOGY JOURNAL



Gambar 25. Notifikasi Telegram yang menunjukan pemblokiran koneksi dari serangan Webshell.

Terlihat pada Gambar 24. merupakan hasil dari serangan *webshell* menggunakan *netcat* yang menujukkan penyerang tidak dapat terhubung ke dalam jaringan *server* di karenakan telah di lakukan pemblockiran.

Berdasarkan serangan-serangan yang telah dilakukan, dapat disimpulkan bahwa server yang telah terintegrasi dengan Wazuh Server akan sangat sulit untuk diserang melalui berbagai metode, mengingat Wazuh Server secara konsisten melakukan pemblokiran terhadap aktivitas-aktivitas yang diidentifikasi sebagai potensi ancaman terhadap server. Selain itu, pada serangan yang telah diuji, terdapat notifikasi yang dikirimkan melalui WhatsApp Bot yang telah dirancang untuk mengirimkan data yang diperoleh dari Wazuh, sehingga memungkinkan pemantauan dan respons yang lebih cepat terhadap ancaman yang terdeteksi. Selain fitur keamanan tersebut, sistem juga dilengkapi dengan mekanisme auto backup yang berfungsi secara berkala untuk memastikan data dan konfigurasi server tetap aman dan dapat dipulihkan dengan mudah dalam situasi darurat.

Hasil ujicoba serangan setelah penerapan *Wazuh* ditunjukkan oleh Tabel 2.

Tabel 2 Jenis Serangan, Deskripsi, Hasil Respon Sistem, dan Notifikasi setelah Penerapan Wazuh

Jenis Serangan	Deskripsi Serangan	Hasil/Respon Sistem	Notifikasi Wazuh
File Vulnerability	Penyerang mengupload file backdoor.php melalui halaman /file.php. File ini bisa memodifikasi dan mengakses file pada server.	File berhasil diupload namun **dihapus secara otomatis sebelum diakses** oleh penyerang.	Terkirim ke Telegram, menunjukkan adanya percobaan file upload.
RemoteCode Execution (RCE)	Penyerang mengakses /web.php dan menjalankan perintah Linux untuk membuat file hacker.php dengan konten berbahaya.	File hacker.php berhasil dibuat tetapi **dihapus secara otomatis** sebelum dapat diakses.	Terkirim ke Telegram, disertai notifikasi dan informasi backup.
Webshell (Netcat)	Penyerang menggunakan netcat untuk mencoba terhubung ke server dengan memanfaatkan file berbahaya (webshell).	Koneksi **tidak berhasil** karena telah dilakukan **pemblokiran** oleh sistem.	Terkirim ke Telegram, menunjukkan upaya koneksi yang diblokir.

3.6. Hasil Analisa

Berdasarkan langkah-langkah uji coba yang telah dilaksanakan, hasil analisis yang diperoleh dapat disimpulkan sebagai berikut :

1. **Sebelum Penerapan Response**
   - Sistem Wazuh belum mampu mencegah serangan *File Upload Vulnerability*, *Remote Code Execution (RCE)*, dan *Webshell*, sehingga semua jenis serangan berhasil dilakukan.
   - Notifikasi Telegram aktif mengirimkan laporan saat serangan terdeteksi, meskipun tidak ada respons otomatis untuk memblokir ancaman.
2. **Setelah Penerapan Response**
   - Wazuh berhasil memblokir seluruh serangan (*File Upload Vulnerability*, *RCE*, dan *Webshell*) dengan efektivitas 100%, sehingga tidak ada serangan yang berhasil dilakukan.
   - Notifikasi Telegram tidak lagi diperlukan karena sistem telah melakukan respons otomatis (*restore direktori*, pemblokiran IP, dan mitigasi proaktif) secara instan.\

IV. PENUTUP

4.1 Kesimpulan

Adapun kesimpulan yang dapat diambil berdasarkan hasil uji coba yang telah dilakukan adalah Implementasi SIEM Wazuh yang terintegrasi dengan Telegram Bot melalui API dari Bot Father menggunakan Bahasa Pemrograman Bash dan *active-response* berhasil mengirimkan data monitoring berbasis JSON ke Telegram, memungkinkan notifikasi real-time terkait aktivitas mencurigakan seperti serangan deface. Wazuh Server terbukti efektif dalam melakukan pemblokiran alamat IP penyerang, merestorasi data yang dimodifikasi, serta mencegah semua serangan (*File Upload Vulnerability*, *Remote Code Execution*, dan *Webshell*) sebelum terjadi modifikasi ilegal pada server. Dashboard Wazuh menyediakan visualisasi data monitoring melalui tabel dan grafik, sementara integrasi dengan Telegram Bot mempercepat respons administrator melalui pemberitahuan langsung. Hasil uji coba menunjukkan wazuh secara konsistn memblokir 100% serangan uji coba via mekanisme *active-response* dalam menangkal serangan, menjadikan solusi ini optimal untuk peningkatan keamanan server berbasis deteksi dini dan mitigasi otomatis.

4.2 Saran

Adapun saran-saran yang dapat diberikan untuk pengembangan SIEM dalam mencengah serangan deface pada server lebih lanjut adalah sebagai berikut: 1. Menambahkan lebih banyak tipe serangan deface pada Wazuh Server. 2. Menambahkan flowchart intertaksi Telegram Bot. 3. Menambahkan lebih banyak active-response yang berhubungan dengan serangan Deface. 4. Memperbaharui atau mengembangkan penelitian ini.

DAFTAR PUSTAKA

[1] S. Sibuea *et al.*, “Clustering perkerjaannya . Salah satu tantangan yang dihadapi oleh perusahaan atau instansi skala besar,” vol. 10, no. 1, pp. 330–345, 2024.

[2] W. P. Putra, R. Burjulius, M. Anis, A. Hilmi, and A. Sumarudin, “Implementasi Sistem Manajemen Log untuk Penanggulangan Serangan Server dengan SIEM”, doi: 10.37817/ikraith-informatika.v8i3.

[3] R. Aditya, Y. Muhyidin, and D. Singasatia, “Implementasi Security Information And Event Management (SIEM) Untuk Monitoring Keamanan Server Menggunakan Wazuh Program Studi Teknik Informatika , Sekolah Tinggi Teknologi Wastukancana , Indonesia penerapan sistem keamanan yang mampu mendeteksi dan men,” vol. 2, no. 5, pp. 137–145, 2024.

[4] D. Lesmideyarti, Q. Hidayati, T. Retno Nugroho, J. Teknik Elektro, J. Perhotelan, and P. Negeri Balikpapan, “Perancangan Infrastruktur dan Implementasi Web Server Untuk Website Sekolah Sebagai Media Informasi dan Komunikasi di SMP PJHI Balikpapan,” 2023.

[5] M. O. Hoshmand, S. Ratnawati, and E. P. Korespondensi, “Analisis Keamanan Infrastruktur Teknologi Informasi dalam Menghadapi Ancaman Cybersecurity,” *J. Sains dan Teknol.*, vol. 5, no. 2, pp. 679–686, 2023, [Online]. Available: https://doi.org/10.55338/saintek.v5i2.2347

[6] M. Hafiz and B. Soewito, “Information Security Systems Design Using SIEM, SOAR and Honeypot,” *J. Pendidik. Tambusai*, vol. 6, no. 2, pp. 15913–15926, 2022, [Online]. Available: https://jptam.org/index.php/jptam/article/view/4898

[7] H. Khotimah, F. Bimantoro, and R. S. Kabanga, “Implementasi Security Information And Event Management (SIEM) Pada Aplikasi Sms Center Pemerintah Daerah Provinsi Nusa Tenggara Barat,” *J. Begawe Teknol. Inf.*, vol. 3, no. 2, 2022, doi: 10.29303/jbegati.v3i2.752.

[8] S. A. Hidayat, “Implementasi Intrusion Detection System Dalam Upaya Pencegahan Cyber Attack,” 2024.

[9] 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,” *J. Teknol. Elekterika*, vol. 20, no. 2, 2023.

[10] E. Aripilahi,) ; Khairil,) ; Abdussalam, and A. Akbar, “Application Of Wazuh To Conduct Monitoring Network Security System (Case Study Of SMK N 1 Bengkulu City) Penerapan Wazuh Untuk Melakukan Monitoring Sistem Keamanan Jaringan (Studi Kasus SMK N 1 Kota Bengkulu).”

[11] W. Abidian and M. A. Setiawan, “Implementasi Splunk dalam Membangun Security Information and Event Management Berdasarkan Log Firewall (studi kasus: Jaringan UII),” *Automata*, 2021, [Online]. Available: https://journal.uii.ac.id/AUTOMATA/article/view/17329%0Ahttps://journal.uii.ac.id/AUTOMATA/article/viewFile/17329/10908

[12] M. A. Fahrudi and I. M. Suartana, “Integrasi

pISSN: 2442-3386 eISSN: 2442-4293
Vol 11 No 1 Jan – Jun 2025, 41 - 42
Melek IT
INFORMATION TECHNOLOGY JOURNAL

End-point Security Berbasis Agent dan Bot Messenger untuk Deteksi dan Monitoring Serangan pada Web Server secara Real-time," *J. Informatics Comput. Sci.*, 2023, doi: 10.26740/jinacs.v4n03.p275-282.

[13] H. Dyan Heluka and W. Sulistyo, "Perancangan Dan Implementasi Security Information and Event Management (SIEM) pada Layanan Virtual Server," *J. Ilm. Komput.*, vol. 19, no. 2, pp. 912–922, 2023.

[14] M. Wahyu, A. S. Fitrani, and H. Hindarto, "Penerapan Bot Telegram untuk Sistem Monitoring Jaringan Intranet Daerah di Instansi Pemerintahan," *Infotek J. Inform. dan Teknol.*, vol. 7, no. 1, pp. 112–122, Jan. 2024, doi: 10.29408/jit.v7i1.24014.

[15] S. P. Rahayu and I. G. L. P. E. Prismana, "Implementasi Monitoring Manajemen Jaringan Dengan Software The Dude Berbasis Telegram Messenger," *J. Informatics Comput. Sci.*, vol. 4, no. 01, pp. 19–25, 2022, doi: 10.26740/jinacs.v4n01.p19-25.

[Halaman ini dibiarkan kosong]