LAPORAN

PRAKTEK KERJA LAPANGAN

IMPLEMENTASI SOAR MENGGUNAKAN WAZUH DI LINGKUNGAN INTERNAL SOC PT TIRTA NAWASENA TEKNOLOGI

Disusun oleh:

BAYU DWI SETIANTO

2207421008

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN

JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER

DEPOK

2025

HALAMAN PENGESAHAN LAPORAN PRAKTIK KERJA LAPANGAN

a. Judul : Implementasi Soar Menggunakan Wazuh di Lingkungan internal SOC PT. Tirta Nawasena Teknologi

b. Penyusun

1) Nama : Bayu Dwi Setianto

2) NIM : 2207421008

c. Program Studi : Teknik Multimedia dan Jaringan

d. Jurusan : Teknik Informatika dan Komputer

e. Waktu Pelaksanaan : 21 Juli 2025 – 31 Desember 2025

f. Tempat Pelaksanaan : PT. TirtaNawasena Teknologi

g. Alamat Pelaksanaan : Jl. Melati I Blok I9 No.6B, Pd. Jagung, Kec.Serpong Utara, Kota Tangerang Selatan, 15323.

Depok, 31 Desember 2025

Pembimbing PNJ

Asep Kurniawan, S.Pd., M.Kom.

NIP. 199109262019031012

Pembimbing Perusahaan

Nasrullah.

Mengesahkan,

KPS Teknik Multimedia dan Jaringan

Dr. Indra Hermawan, S.Kom., M.Kom.

NIP. 198703132019031012

KATA PENGANTAR

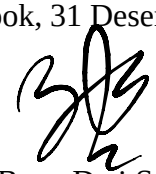
Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena atas berkat dan rahmat-Nya penulis dapat menyelesaikan Laporan Praktik Kerja Lapangan (PKL) ini dengan baik. Penyusunan laporan ini merupakan salah satu syarat untuk memenuhi kelulusan pada Program Diploma Empat Politeknik Negeri Jakarta.

Penulis menyadari bahwa tanpa dukungan, bantuan, dan bimbingan dari berbagai pihak sejak masa perkuliahan hingga proses penyusunan laporan ini, sangatlah sulit bagi penulis untuk menyelesaikan laporan ini. Oleh karena itu, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

a. Bpk. Asep Kurniawan, S.Pd., M.Kom, selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan penulis dalam penyusunan laporan Praktik Kerja Lapangan ini;
b. Dr. Indra Hermawan, S.Kom., M.Kom., selaku Kepala Program Studi Teknik Multimedia dan Jaringan yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan penulis dalam penyusunan laporan Praktik Kerja Lapangan ini;
c. Pihak PT. Tirta Nawasena Teknologi Divisi SOC (*Security Operation Center*) khususnya kepada Pak Nasrullah selaku mentor saya selama magang dan juga Mas Franz Davidson yang telah banyak membantu dalam usaha memperoleh data yang penulis perlukan;
d. Orang tua dan keluarga saya yang telah memberikan bantuan dukungan moral dan material;
e. Sahabat dan rekan-rekan seperjuangan dari TMJ Angkatan 2022 yang telah membantu dalam penyusunan laporan Praktik Kerja Lapangan ini.

Akhir kata, penulis berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga laporan Praktik Kerja Lapangan ini membawa manfaat bagi pengembangan ilmu.

Depok, 31 Desember 2025



Bayu Dwi Setianto

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi informasi yang semakin pesat telah mendorong hampir seluruh sektor industri untuk beralih menuju digitalisasi proses operasional. Sistem komunikasi, layanan pelanggan, manajemen data, hingga infrastruktur bisnis kini bergantung pada perangkat dan jaringan digital. Namun, transformasi digital tersebut diikuti oleh meningkatnya risiko serangan siber yang semakin canggih dan beragam, seperti brute-force attack, malware, ransomware, phishing, hingga eksploitasi celah keamanan sistem. Serangan-serangan tersebut dapat mengancam kerahasiaan, integritas, dan ketersediaan data perusahaan, serta menyebabkan kerugian finansial maupun operasional.

Untuk menghadapi ancaman tersebut, banyak organisasi membentuk *Security* Operation Center (SOC) yang berfungsi melakukan pemantauan, deteksi, analisis, dan penanganan insiden keamanan secara terpusat dan real-time. Dalam operasionalnya, SOC memanfaatkan sistem SIEM (*Security Information and Event Management*) untuk mengumpulkan serta menganalisis log dari berbagai perangkat, serta SOAR (*Security Orchestration, Automation, and Response*) untuk melakukan otomatisasi langkah respons terhadap ancaman.

Namun, pada praktiknya, proses penanganan insiden di SOC masih sering dilakukan secara manual oleh analis keamanan. Ketika volume alert tinggi, proses manual ini dapat menyebabkan keterlambatan dalam merespons insiden, meningkatnya beban kerja analis, serta potensi terjadinya human error. Hal ini diperkuat oleh laporan terbaru dari IBM *Security* (2024) yang menyatakan bahwa organisasi yang menerapkan keamanan AI dan otomatisasi secara ekstensif mampu menyelesaikan insiden 98 hari lebih cepat dibandingkan organisasi yang tidak menerapkannya. Data tersebut menunjukkan bahwa otomatisasi respons keamanan bukan lagi sekadar pilihan, melainkan kebutuhan krusial bagi organisasi modern untuk meminimalisasi dampak serangan(IBM Security, 2024)

Sebagai upaya meningkatkan efektivitas SOC, salah satu solusi yang diterapkan adalah memanfaatkan teknologi SOAR. Dalam konteks ini, Wazuh, sebuah

platform keamanan berbasis open-source yang menggabungkan fungsi SIEM dan XDR, dipilih karena menyediakan fitur Active Response. Fitur ini mampu melakukan respons otomatis terhadap ancaman yang terdeteksi, seperti memblokir alamat IP berbahaya atau menghentikan proses mencurigakan secara instan tanpa intervensi langsung dari analis, sehingga waktu respons dapat dipangkas secara signifikan.

PT Tirta Nawasena Teknologi, sebagai perusahaan yang terus berkembang, menyadari pentingnya langkah preventif dalam menjaga keamanan infrastrukturnya. Oleh karena itu, dalam kegiatan magang ini, penulis menjalankan sebuah proyek inisiatif untuk memperkuat pertahanan siber di lingkungan **Internal SOC perusahaan**. Proyek ini difokuskan pada implementasi SOAR menggunakan Wazuh untuk mengamankan lingkungan internal di masa depan. Kegiatan ini meliputi instalasi, konfigurasi, serta pengujian skenario serangan pada infrastruktur internal untuk memastikan mekanisme otomatisasi respons berjalan efektif sebelum diterapkan pada skala yang lebih luas. Implementasi ini diharapkan mampu meningkatkan ketahanan siber (cyber resilience) serta menjadi standar baru dalam efisiensi operasional internal perusahaan.

Oleh karena itu, laporan ini disusun untuk menjelaskan proses implementasi SOAR menggunakan Wazuh pada lingkungan internal SOC PT Tirta Nawasena Teknologi, sekaligus menjadi dokumentasi ilmiah atas inisiatif, pengalaman, dan hasil pengujian yang diperoleh selama pelaksanaan magang.

1.2. Ruang Lingkup Kegiatan

Ruang lingkup kegiatan Magang ini berfokus pada proses perancangan, implementasi, dan pengujian sistem otomatisasi respons keamanan (*Security Orchestration, Automation, and Response - SOAR*) menggunakan Wazuh di lingkungan Internal SOC PT Tirta Nawasena Technology.

Adapun batasan dan rincian kegiatan meliputi:

1. **Instalasi dan Konfigurasi Infrastruktur** Melakukan instalasi Wazuh

Manager dan Wazuh Agent, serta mengintegrasikannya dengan mesin deteksi eksternal (Suricata, ClamAV, YARA, dan VirusTotal) pada lingkungan simulasi SOC internal yang bersifat lokal (on-premise) dan terisolasi.

2. **Implementasi Active Response untuk Mitigasi Jaringan** Menerapkan mekanisme respons otomatis untuk menangani serangan jaringan, khususnya pemblokiran alamat IP (IP Blocking) secara real-time melalui manipulasi aturan firewall (iptables) ketika terdeteksi aktivitas brute-force atau pemindaian agresif.

3. **Implementasi Active Response untuk Mitigasi Malware** Membangun mekanisme pertahanan berlapis (*tiered security*) untuk menangani malicious file. Hal ini mencakup deteksi otomatis dan eksekusi tindakan karantina atau penghapusan file berbahaya menggunakan logika integrasi antara ClamAV, YARA, dan validasi hash VirusTotal.

4. **Pengujian Serangan Terkontrol** Melakukan simulasi serangan untuk memvalidasi efektivitas sistem, yang difokuskan pada dua skenario utama:

    a. **Skenario Jaringan**: Simulasi SSH Brute-force dan Port Scanning untuk memicu pemblokiran IP.

    b. **Skenario Endpoint**: Simulasi penyisipan sampel malware (seperti EICAR) dan file dengan pola string berbahaya untuk memicu karantina otomatis.

5. **Batasan Implementasi** Kegiatan ini tidak mencakup penerapan pada infrastruktur produksi (live production), integrasi dengan SIEM eksternal perusahaan, maupun penanganan insiden siber yang berada di luar skenario pengujian yang telah ditentukan.

Dengan ruang lingkup tersebut, kegiatan magang ini diharapkan mampu memberikan gambaran teknis yang komprehensif mengenai penerapan SOAR berbasis Wazuh untuk penanganan malicious file dan serangan brute-force, serta menjadi dasar pengembangan otomatisasi keamanan di SOC PT Tirta Nawasena

Teknologi pada masa mendatang.

1.3. Waktu dan Tempat Pelaksanaan

Tempat dan pelaksanaan	: Jl. Melati I Blok I9 No.6B, Pd. Jagung, Kec. Serpong Utara, Kota Tangerang Selatan, Banten 15323.
Penempatan	: Departement SOC (*Security Operation Center*) – SOC *Analyst*.
Waktu Pelaksanaan	: 21 Juli – 30 Desember 2025.
Lama PKL	: 6 Bulan.

1.4. Tujuan

Tujuan dari implementasi SOAR menggunakan Wazuh di lingkungan internal SOC PT Tirta Nawasena Technology, adalah sebagai berikut:

1. Mengimplementasikan sistem keamanan terorkestrasi (*Security Orchestration, Automation, and Response* - SOAR) berbasis Wazuh sebagai solusi otomatisasi respons insiden di lingkungan Internal SOC.

2. Menguji dan menganalisis kinerja fitur Active Response dalam mendeteksi serta menanggulangi serangan SSH brute-force dan ancaman malicious file secara otomatis tanpa intervensi manual.

1.5.Kegunaan

Kegunaan dari implementasi SOAR menggunakan Wazuh di lingkungan Internal SOC ini adalah sebagai berikut:

1. **Tersedianya Solusi Otomatisasi Respons di Internal SOC** Menghasilkan sistem SOAR berbasis Wazuh yang siap beroperasi di lingkungan Internal SOC, berguna untuk mengubah mekanisme penanganan insiden yang sebelumnya manual menjadi terotomatisasi, sehingga meningkatkan efisiensi operasional keamanan secara menyeluruh.pada prosedur manual yang memakan waktu.

2. **Terbuktinya Efektivitas Mitigasi Ancaman Tanpa Intervensi** Memberikan bukti teknis mengenai kemampuan fitur Active Response dalam mendeteksi dan menanggulangi serangan SSH brute-force serta malicious file secara instan, yang berguna untuk memvalidasi keandalan sistem dalam melindungi aset jaringan dan endpoint tanpa menunggu tindakan manusia.

BAB II LANDASAN TEORI

2.1.*Security Information and Event Management* (SIEM)

Security Information and Event Management (SIEM) adalah pendekatan komprehensif dalam manajemen keamanan siber yang menggabungkan manajemen informasi keamanan *(Security Information Management* - SIM) dan manajemen peristiwa keamanan *(Security Event Management* - SEM). Sistem ini berfungsi untuk mengumpulkan, menganalisis, dan menyajikan informasi keamanan dari berbagai perangkat jaringan dan aplikasi secara terpusat.(Jurnal *et al.*, 2024)

Menurut NIST (National Institute of Standards and Technology) dalam publikasi khususnya, teknologi SIEM menyediakan analisis real-time dari peringatan keamanan yang dihasilkan oleh aplikasi dan perangkat keras jaringan. Kemampuan utamanya meliputi agregasi log, korelasi kejadian (event correlation), peringatan otomatis, dan dasbor kepatuhan.(Kent and Souppaya, no date)

2.2.*Security Orchestration, Automation, and Response* (SOAR)

SOAR didefinisikan sebagai kumpulan teknologi yang memungkinkan organisasi untuk mengumpulkan data ancaman keamanan dari berbagai sumber dan merespons kejadian keamanan tingkat rendah secara otomatis tanpa campur tangan manusia.(Aljahdali and Alsulami, 2025)

Berbeda dengan SIEM yang berfokus pada deteksi, SOAR berfokus pada tindakan (action). Gartner, yang mempopulerkan istilah ini, menjelaskan bahwa SOAR menggabungkan orkestrasi keamanan, otomatisasi, dan respons insiden dalam satu platform untuk meningkatkan efisiensi operasional Security Operation Center (SOC).(*3 Must-Haves in Your Cybersecurity Incident Response Planning End-to-End Incident Response-Before It's Needed 3 Must-Haves in Your Cybersecurity Incident Response*, no date)

2.3.Wazuh

Wazuh adalah platform keamanan open-source yang menyediakan perlindungan XDR (Extended Detection and Response) dan SIEM terpadu. Wazuh lahir sebagai

fork dari OSSEC dan telah berkembang menjadi solusi yang mencakup pemantauan integritas file, deteksi kerentanan, analisis log, dan respons aktif.

Arsitektur Wazuh terdiri dari Wazuh Agent yang dipasang pada endpoint (server/komputer target) dan Wazuh Manager yang bertugas mengumpulkan dan menganalisis data. Wazuh juga menggunakan Indexer dan Dashboard untuk visualisasi data.(Wazuh Inc., 2025)

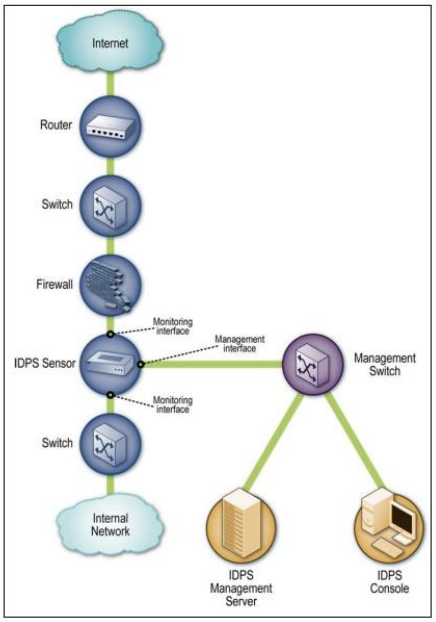
Gambar 2. 1 Logo Wazuh

Sumber (https://wazuh.com/uploads/2022/05/Logo-blogpost.png)

2.4.Network Intrusion Detection System (NIDS)

Network Intrusion Detection System (NIDS) adalah sistem keamanan yang ditempatkan pada titik strategis dalam jaringan untuk memantau lalu lintas (traffic) menuju dan dari semua perangkat di jaringan. NIDS menganalisis lalu lintas untuk mencari tanda-tanda serangan atau perilaku abnormal.

Menurut Scarfone dan Mell (NIST), NIDS bekerja dengan cara menyalin paket data yang lewat (promiscuous mode) dan membandingkannya dengan database tanda tangan serangan (signature-based) atau model perilaku normal (anomaly-based).(Scarfone and Mell, 2007)



Gambar 2. 2 Contoh Inline Network-Based IDPS Sensor Architecture

2.5.Malware

Malware (singkatan dari Malicious Software) adalah istilah umum yang merujuk pada perangkat lunak apa pun yang dirancang untuk menyebabkan kerusakan pada komputer, server, klien, atau jaringan komputer. Jenis malware meliputi virus, worm, trojan, ransomware, spyware, dan adware.

Menurut National Institute of Standards and Technology (NIST) dalam publikasi khususnya, malware didefinisikan sebagai program yang disisipkan ke dalam sistem, biasanya secara diam-diam, dengan tujuan mengkompromikan kerahasiaan, integritas, atau ketersediaan data, aplikasi, atau sistem operasi korban.(Souppaya and Scarfone, 2013)

2.6.VirusTotal

VirusTotal adalah layanan daring gratis yang menganalisis berkas dan URL (tautan) untuk mendeteksi virus, worm, trojan, dan jenis konten berbahaya lainnya. VirusTotal bekerja sebagai agregator informasi yang menggunakan lebih dari 70 mesin antivirus dan pemindai URL.

Dalam konteks SOAR, VirusTotal sering digunakan sebagai Threat Intelligence Source untuk melakukan pengayaan data (enrichment). Ketika sebuah hash file mencurigakan ditemukan, sistem dapat menanyakan VirusTotal untuk

memverifikasi apakah file tersebut berbahaya berdasarkan konsensus vendor keamanan global.(VirusTotal, 2024)

Gambar 2. 3 Logo Virustotal

Sumber (https://upload.wikimedia.org/wikipedia/commons/thumb/b/b7/VirusTotal_logo.svg/2560px-VirusTotal_logo.svg.png)

2.7.File Integrity Monitoring (FIM)

File Integrity Monitoring (FIM) adalah proses keamanan internal yang memvalidasi integritas file sistem operasi dan aplikasi dengan cara memverifikasi file saat ini terhadap baseline yang diketahui baik. FIM bekerja dengan menghitung checksum (hash kriptografis seperti MD5 atau SHA-256) dari file penting. Jika hash berubah, berarti file telah dimodifikasi.

FIM sangat penting untuk mendeteksi serangan yang mengubah file konfigurasi sistem atau menyisipkan backdoor.(Nel and Khiabani, 2021)

Gambar 2. 4 Definisi FIM

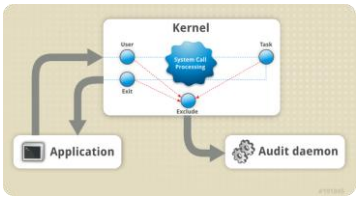
Sumber (https://www.bitlyft.com/hs-fs/hubfs/FIM.jpg?width=3072&height=1276&name=FIM.jpg)

2.8.Auditd

Linux Audit Daemon (Auditd) adalah subsistem audit tingkat kernel pada sistem operasi Linux. Auditd bertanggung jawab untuk menulis catatan audit (audit records) ke disk. Alat ini dapat melacak peristiwa keamanan sistem, mencatat akses file, eksekusi perintah, dan panggilan sistem (syscalls).

Dalam arsitektur Wazuh, Auditd sering digunakan untuk memperkaya data FIM dengan informasi "siapa" yang melakukan perubahan file (atribut whodata).(Red

Hat, 2025)



Gambar 2. 5 System Auditing

Sumber (https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/6/html/security_guide/chap-system_auditing)

2.9.YARA

YARA adalah alat yang ditujukan (namun tidak terbatas) untuk membantu peneliti malware dalam mengidentifikasi dan mengklasifikasikan sampel malware. YARA bekerja berdasarkan aturan (rules) yang dibuat untuk mencari pola string atau biner tertentu di dalam sebuah file.(VirusTotal Revision, 2022)

YARA sering disebut sebagai "pisau lipat Swiss" untuk pencocokan pola (pattern matching). Dengan YARA, analis keamanan dapat membuat aturan kustom untuk mendeteksi serangan yang ditargetkan yang mungkin tidak terdeteksi oleh antivirus standar.

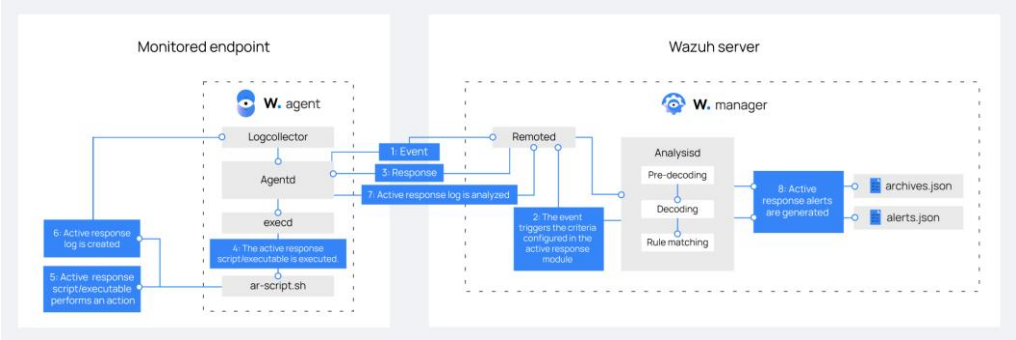
Gambar 2. 6 Logo Yara

Sumber (https://miro.medium.com/v2/resize:fit:1400/1*CkKv6zWhxXlHgGWPedV6SQ.jpeg)

2.10. Active Response

Active Response adalah mekanisme otomatis yang memicu tindakan balasan sebagai respons terhadap peringatan keamanan tertentu. Dalam konteks Wazuh, *Active Response* menjalankan skrip (misalnya: memblokir IP di firewall, menghapus file berbahaya, atau mengisolasi endpoint) ketika aturan deteksi tertentu terpenuhi.(Wazuh Inc, 2025)

Fitur ini merupakan komponen kunci dalam membangun sistem SOAR sederhana menggunakan Wazuh, karena mengubah deteksi pasif menjadi pencegahan aktif.



Gambar 2. 7 Diagram alur Active Response

Sumber (https://documentation.wazuh.com/current/_images/active-response-workflow1.png)

2.11. Suricata

Suricata adalah mesin *Network Threat Detection* berkinerja tinggi, matang, dan bersifat open-source. Suricata berfungsi sebagai IDS *(Intrusion Detection System)*, IPS (I*ntrusion Prevention System)*, dan pemantauan keamanan jaringan *(Network Security Monitoring).(OISF, 2025)*

Keunggulan Suricata dibandingkan pendahulunya (seperti Snort single-threaded lama) adalah arsitektur multi-threaded yang memungkinkannya menangani lalu lintas jaringan bervolume tinggi secara efisien. Suricata menggunakan bahasa aturan (rule language) untuk mendeteksi ancaman yang diketahui.

Gambar 2. 8 Logo Suricata

Sumber (https://suricata.io/wp-content/uploads/2023/09/Suricata_logo_600x600-1.png)

2.12. Docker

Docker adalah platform terbuka untuk mengembangkan, mengirimkan, dan menjalankan aplikasi. Docker memungkinkan pemisahan aplikasi dari infrastruktur melalui teknologi kontainerisasi. Kontainer Docker membungkus aplikasi beserta dependensinya (pustaka, file konfigurasi) ke dalam satu unit standar yang dapat dijalankan di lingkungan apa pun.(Docker Inc, 2025)

Dalam konteks keamanan siber, Docker sering digunakan untuk menerapkan alat keamanan (seperti Wazuh Manager dan Suricata) dengan cepat, terisolasi, dan

mudah direplikasi (reproducible).

Gambar 2. 9 Logo Docker

Sumber (https://perfectmediaserver.com/images/logos/docker-logo-horizontal.png)

2.13. Extended Detection and Response (XDR)

Extended Detection and Response (XDR) adalah pendekatan keamanan siber terpadu yang memberikan visibilitas holistik dan kontrol di seluruh infrastruktur teknologi informasi, termasuk endpoint, jaringan, cloud, dan beban kerja server. Berbeda dengan solusi keamanan tradisional yang bekerja secara terpisah, XDR secara otomatis mengumpulkan dan mengkorelasikan data telemetri dari berbagai lapisan keamanan untuk mendeteksi ancaman yang lebih kompleks dengan lebih cepat dan akurat(Microsoft, 2025).

Dalam konteks sistem yang dibangun, konsep XDR diterapkan melalui integrasi antara Wazuh (sebagai pusat analisis dan *endpoint security*) dan Suricata (sebagai sensor jaringan). Penggabungan ini memungkinkan sistem tidak hanya melihat ancaman dari satu sisi, melainkan merespons serangan secara otomatis di kedua lapisan (memblokir IP di *firewall* jaringan dan mengkarantina *file* di *endpoint).*

2.14. Flowchart

Flowchart atau diagram alur adalah representasi visual dari urutan langkah-langkah dan keputusan yang diperlukan untuk melakukan suatu proses. Setiap langkah dalam urutan digambarkan dalam bentuk simbol diagram standar (seperti oval untuk terminal, persegi panjang untuk proses, dan belah ketupat untuk keputusan) yang dihubungkan dengan garis panah untuk menunjukkan arah alur proses logis(SmartDraw, no date) .

Dalam perancangan sistem SOAR ini, flowchart digunakan untuk memetakan logika algoritma pada skrip respons otomatis (seperti integrated_scan.sh dan firewall-drop.sh). Diagram ini membantu memvisualisasikan bagaimana sistem

mengambil keputusan "Ya/Tidak" saat mendeteksi anomali sebelum mengeksekusi tindakan mitigasi.

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

Gambar 2. 10 Contoh Flowchart

Sumber (https://www.smartdraw.com/flowchart/img/basic-symbols-table.jpg)

2.15. Serangan Brute-Force

Serangan Brute-Force adalah metode peretasan di mana penyerang mencoba mendapatkan akses tidak sah ke suatu sistem dengan cara menebak kredensial (username dan password) secara berulang-ulang menggunakan metode trial and error. Penyerang biasanya menggunakan perangkat lunak otomatis untuk mencoba jutaan kombinasi kata sandi dalam waktu singkat hingga menemukan kombinasi yang benar, sering kali menargetkan protokol seperti SSH atau RDP(owasp, 2023).

Dalam penelitian ini, serangan Brute-Force menjadi salah satu skenario pengujian utama terhadap efektivitas NIDS Suricata. Sistem dikonfigurasi untuk mengenali lonjakan percobaan login yang gagal dalam waktu singkat dan secara otomatis memicu Active Response untuk memblokir alamat IP penyerang sebelum akses berhasil didapatkan.

BAB III HASIL PELAKSANAAN MAGANG

3.1.Profil DUDI

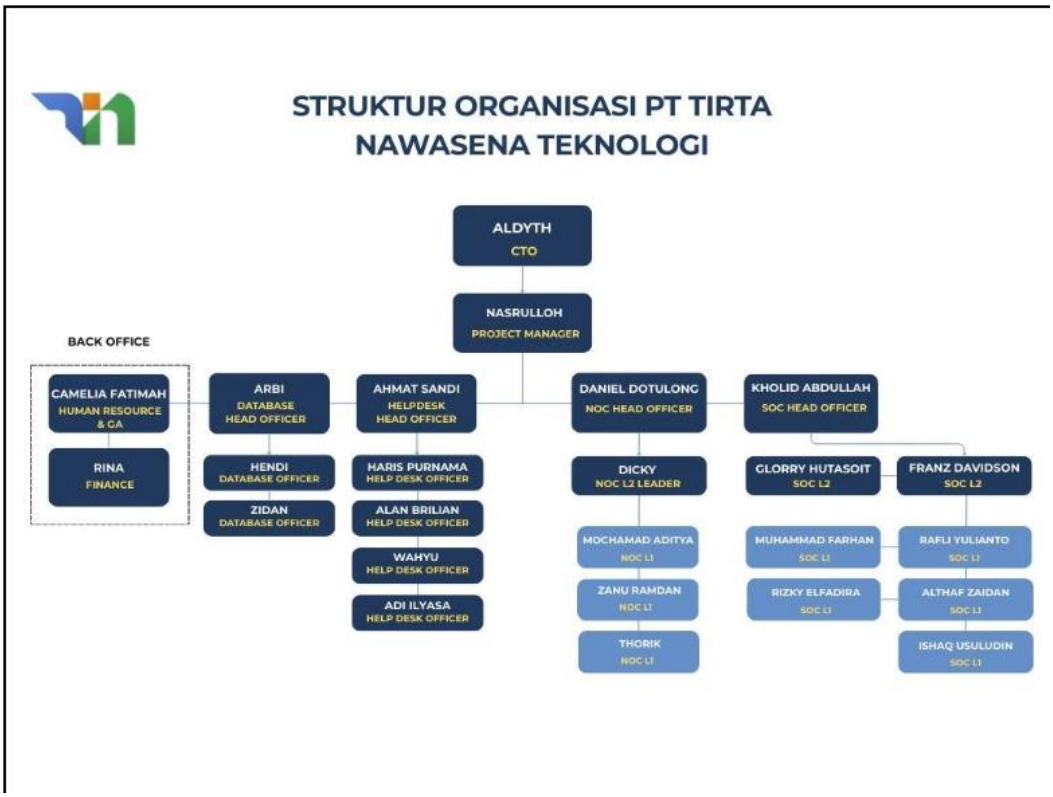
Gambar 3. 1 Logo PT Tirta Nawasena Teknologi

Sumber (https://tirtanawasena.com/)

PT Tirta Nawasena Teknologi merupakan perusahaan penyedia solusi teknologi informasi (IT) *End-to-End* dan Manajemen Fasilitas yang berorientasi pada aspek keamanan serta teknologi operasional. Perusahaan ini berdomisili di Jl. Melati I Blok I9 No.6B, Pd. Jagung, Kec. Serpong Utara, Kota Tangerang Selatan, Banten 15323.

Dalam menjalankan operasionalnya, PT Tirta Nawasena Teknologi telah memenuhi standar internasional yang dibuktikan dengan sertifikasi ISO 37001:2016 (*Anti-Bribery Management System*) dan ISO 9001:2015 (*Quality Management System*). Kredibilitas perusahaan juga tercermin dari kemitraan strategis yang terjalin dengan berbagai instansi pemerintah (BUMN) maupun sektor swasta, seperti PLN, State Bank India (SBI), Semen Indonesia Group (SIG), Badan Kepegawaian Negara (BKN), Kredit Biro Indonesia Jaya (KBIJ), dan Waskita.

Fokus bisnis utama perusahaan terletak pada ranah *operational security* dan *managed services.* Layanan yang ditawarkan mencakup manajemen keamanan terintegrasi, *cyber security*, pemantauan sistem, serta pengelolaan infrastruktur teknologi bagi organisasi berskala besar. Berbeda dengan perusahaan pengembang produk perangkat lunak, kekuatan PT Tirta Nawasena Teknologi terletak pada implementasi, pengawasan, tata kelola IT, manajemen risiko, serta pemeliharaan keberlangsungan sistem (*maintenance*) bagi klien korporat maupun pemerintahan.



Gambar 3. 2 Struktur organisasi PT Tirta Nawasena Teknologi

Berdasarkan struktur organisasi yang disajikan pada Gambar 3.2, penulis ditempatkan pada divisi *Security Operations Center* (SOC) Level 1. Posisi ini bertindak sebagai garda terdepan *(first line of defense)* dalam operasional keamanan siber perusahaan.

Penulis bertanggung jawab untuk melakukan analisis awal terhadap anomali atau indikasi ancaman, mengklasifikasikan insiden, serta membuat laporan eskalasi ke tingkat lanjutan jika terdeteksi adanya potensi serangan siber. Melalui penempatan ini, penulis memperoleh pemahaman mendalam mengenai prosedur standar respons insiden, analisis log, serta operasional keamanan di lingkungan perusahaan yang sebenarnya.

3.2.Uraian Kegiatan Magang

Pelaksanaan kegiatan magang dilakukan secara penuh di kantor (*Work From Office*) dengan sistem dua *shift*, yaitu pukul 08.00–16.00 WIB dan 15.00–00.00 WIB. Tiga peserta magang dibagi ke dalam kedua *shift* tersebut dengan rotasi mingguan, sehingga setiap peserta memperoleh pengalaman bekerja pada kedua rentang waktu operasional. Seluruh aktivitas harian dicatat dalam log kegiatan

SOC *Level* 1 sebagai bagian dari proses evaluasi.

Pada bulan pertama (Juli 2025), kegiatan berfokus pada proses orientasi lingkungan kerja dan pengenalan perangkat SOC. Peserta mempelajari platform Crowdstrike sebagai alat utama pemantauan *endpoint*, khususnya modul *Endpoint Detection*. Analisis awal dilakukan terhadap indikator seperti hash SHA-256, domain, dan IP, dengan penilaian *alert* mengacu pada informasi *Cyber Threat Intelligence* (CTI) untuk klasifikasi insiden.

Memasuki bulan kedua (Agustus 2025), volume pekerjaan meningkat dengan tugas verifikasi *quarantine* data domain menggunakan sumber CTI seperti Cyfirma dan VirusTotal. Peserta juga mengikuti pelatihan Fortinet NSE 1. Pada minggu terakhir bulan ini, peserta mulai diperkenalkan dengan konsep SOAR (*Security Orchestration, Automation, and Response*) dan mulai melakukan instalasi awal Wazuh Manager sebagai fondasi proyek akhir.

Pada bulan ketiga (September 2025), selain rutin memantau *honeypot* dan data breach via Tehtris, fokus utama beralih pada perancangan arsitektur sistem keamanan terpadu. Kegiatan meliputi pembuatan *flowchart* alur kerja SOAR, konfigurasi dasar Wazuh, serta instalasi Suricata sebagai NIDS. Peserta juga mulai mengembangkan skrip *Active Response* awal untuk merespons ancaman jaringan secara otomatis, serta mengintegrasikan log Suricata agar dapat diteruskan ke Wazuh Manager.

Pada bulan keempat (Oktober 2025), kegiatan difokuskan pada pengujian intensif dan *troubleshooting*. Peserta melakukan simulasi serangan (seperti *port scanning* dan *brute-force*) untuk memvalidasi efektivitas aturan deteksi. Tahapan ini mencakup perbaikan *decoder* Wazuh untuk membaca log Suricata (eve.json), konfigurasi *File Integrity Monitoring* (FIM), serta penyempurnaan mekanisme *Active Response* agar dapat memblokir IP penyerang secara otomatis di *firewall*. Visualisasi data serangan juga mulai dibangun pada Wazuh Dashboard.

Pada bulan kelima (November 2025), dilakukan pengembangan fitur lanjutan untuk memperkaya kemampuan deteksi sistem. Implementasi meliputi integrasi YARA untuk deteksi *malware* berbasis pola, pengaktifan modul *Vulnerability Detector* untuk memindai kerentanan *endpoint*, serta integrasi API VirusTotal dan AlienVault OTX untuk pengayaan data intelijen otomatis. Peserta juga menyusun

mekanisme notifikasi insiden dan strategi *backup* konfigurasi sistem.

Pada bulan terakhir (Desember 2025), kegiatan *monitoring* rutin Crowdstrike dan Tehtris tetap berjalan beriringan dengan finalisasi proyek Wazuh. Fokus utama adalah evaluasi fungsional sistem secara keseluruhan untuk memastikan penurunan MTTR *(Mean Time To Respond),* audit log *Active Response*, serta penyusunan laporan teknis. Kegiatan ditutup dengan serah terima *(handover)* dokumentasi topologi dan konfigurasi sistem kepada tim internal perusahaan serta penyelesaian laporan akhir magang. Rincian lengkap mengenai log aktivitas harian selama periode magang dapat dilihat pada Lampiran 7.

3.3.Pembahasan Hasil Magang

3.3.1. Gambaran Umum Sistem

Sistem yang dibangun merupakan sebuah platform keamanan terpadu berbasis *Security Orchestration, Automation, and Response* (SOAR) yang dirancang untuk lingkungan Internal SOC. Arsitektur sistem ini menggunakan Wazuh sebagai inti SIEM (*Security Information and Event Management*) yang berfungsi mengagregasi data, mengelola korelasi event, dan mengorkestrasi respons otomatis. Sistem ini mengintegrasikan berbagai sensor keamanan eksternal untuk membentuk mekanisme pertahanan yang komprehensif, sebagaimana diilustrasikan pada desain arsitektur sistem.

Secara teknis, sistem menerapkan strategi Pertahanan Berlapis (*Defense in Depth*) yang mencakup dua domain utama:

a. **Keamanan *Endpoint* (*Host-Based*):** Pada sisi agent, sistem menerapkan mekanisme deteksi bertingkat (*tiered security*) untuk memaksimalkan akurasi dan efisiensi sumber daya. Deteksi dimulai dari pemantauan level kernel menggunakan Auditd dan perubahan *file* melalui FIM. Analisis malware dilakukan secara berjenjang, dimulai dari pemindaian *signature* lokal oleh ClamAV, dilanjutkan dengan pencocokan pola konten oleh YARA, dan diakhiri dengan validasi reputasi hash berbasis *cloud* menggunakan VirusTotal API untuk mengenali ancaman *zero-day*.

b. **Keamanan Jaringan (*Network-Based*):** Di sisi jaringan, Suricata difungsikan

sebagai NIDS (*Network Intrusion Detection System*) yang memantau lalu lintas keluar-masuk secara *real-time*. Suricata bertugas mendeteksi anomali *trafik* seperti pemindaian port (*port scanning*), upaya *brute-force*, dan pola serangan jaringan lainnya, yang kemudian log-nya diteruskan ke Wazuh Manager.

Sistem ini dirancang khusus untuk mengatasi kendala operasional SOC tradisional, seperti fragmentasi alat (*tool fragmentation*) dan tingginya waktu respons manual. Seluruh log aktivitas dikirim ke Wazuh Manager untuk dikorelasikan menjadi satu peringatan (*alert*) yang bermakna. Hal ini memberikan analis SOC visibilitas menyeluruh (*holistic visibility*) melalui Wazuh Dashboard, yang mempermudah proses investigasi insiden.

Fitur unggulan dari arsitektur ini adalah implementasi Active Response yang terorkestrasi. Sistem tidak hanya mendeteksi ancaman, tetapi juga mampu merespons secara otonom (otomatis). Mekanisme ini mencakup tindakan karantina file secara instan saat terdeteksi malware di *endpoint*, serta pemblokiran alamat IP penyerang pada firewall saat terdeteksi serangan jaringan. Sinergi antara Wazuh, Suricata, dan integrasi pihak ketiga ini menciptakan ekosistem keamanan yang proaktif, responsif, dan mampu meminimalisir waktu penanganan insiden (*Mean Time to Respond*) secara signifikan.

3.3.2. Analisis Kebutuhan

Agar sistem *Security Orchestration, Automation, and Response* (SOAR) yang dirancang dapat beroperasi secara optimal di lingkungan Internal SOC, diperlukan identifikasi kebutuhan yang komprehensif. Analisis ini dibagi menjadi kebutuhan fungsional, non-fungsional, serta kebutuhan perangkat keras dan lunak.

a. **Kebutuhan Fungsional** Kebutuhan fungsional mendefinisikan fitur dan layanan spesifik yang harus disediakan oleh sistem untuk mencapai tujuan otomatisasi keamanan:

1) Sentralisasi dan Korelasi Log (SIEM):

a. Sistem harus mampu mengumpulkan log (*log collector*) dari berbagai *endpoint* dan sensor jaringan secara real-time.

b. Sistem harus mampu melakukan korelasi event untuk menghubungkan aktivitas terpisah menjadi satu indikator serangan yang valid.

2) Deteksi Ancaman Jaringan (NIDS):

Sistem harus mampu memantau lalu lintas jaringan dan mendeteksi pola serangan berbasis jaringan seperti port scanning, brute-force, dan upaya eksploitasi protokol menggunakan Suricata.

3) Deteksi dan Validasi Ancaman *Endpoint* (Defense in Depth):

a. Sistem harus memantau integritas file penting (File Integrity Monitoring/FIM) untuk mendeteksi perubahan yang tidak sah.

b. Sistem harus mampu memindai malware menggunakan metode berbasis signature (ClamAV) dan pencocokan pola konten (YARA).

c. Sistem harus terintegrasi dengan layanan Threat Intelligence eksternal (VirusTotal API) untuk memvalidasi hash file mencurigakan (enrichment).

4) Otomatisasi Respons (Active Response):

Sistem harus mampu mengeksekusi tindakan mitigasi otomatis tanpa intervensi manusia, meliputi:

a. **Pemblokiran Jaringan**: Memutus koneksi atau memblokir IP penyerang pada firewall saat terdeteksi serangan jaringan.

b. **Karantina File**: Memindahkan atau menghapus file berbahaya secara otomatis saat terdeteksi oleh modul antivirus/YARA.

5) Visualisasi dan Pelaporan:

Sistem harus menyediakan dashboard interaktif untuk memvisualisasikan status keamanan, daftar alert, dan riwayat respons yang telah dieksekusi.

b. **Kebutuhan Non-Fungsional** Kebutuhan ini berkaitan dengan batasan

operasional dan kualitas layanan sistem:

1) **Kinerja (Performance):** Sistem harus mampu memproses log dan mengeksekusi Active Response dengan latensi rendah (mendekati real-time) untuk meminimalisir dampak serangan.

2) **Interoperabilitas:** Sistem harus dapat berjalan di atas kontainerisasi (Docker) dan mendukung integrasi antara komponen berbeda (Wazuh, Suricata, ClamAV) tanpa konflik.

3) **Efisiensi Sumber Daya:** Mengingat batasan API gratis (VirusTotal), sistem harus memiliki logika efisiensi (mekanisme *tiered security*) untuk mencegah penggunaan kuota berlebih.

c. **Kebutuhan Perangkat Keras** dan Lunak Untuk mengimplementasikan arsitektur tersebut, dibutuhkan spesifikasi lingkungan sebagai berikut:

1) Perangkat Keras (Server & *Endpoint* Simulasi):

Tabel 3. 1 Spesifikasi Kebutuhan Perangkat Keras

PERANGKAT	CPU	RAM	STORAGE
Suricata Server (Wazuh Docker)	8 vCPU	8 GB	150 GB
Endpoint/Agent	4 vCPU	2 GB	25 GB
Kali Linux (Untuk Pengujian)	4 vCPU	2 GB	25 GB

2) Perangkat Lunak:

a) **Sistem Operasi**: Linux Ubuntu Server (sebagai Host Wazuh & Suricata) dan OS target (Linux/Windows) sebagai *Endpoint*.

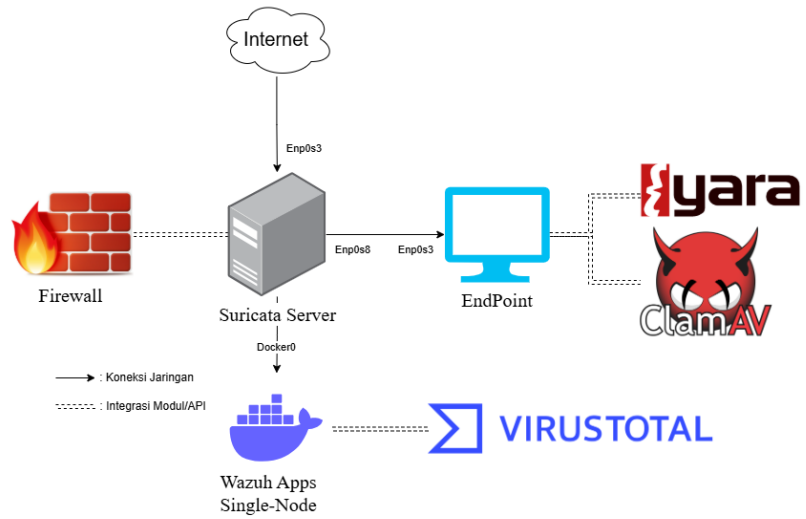
b) **Platform Kontainer**: Docker & Docker Compose (untuk deployment Wazuh Manager).

c) **Engine Keamanan**: Wazuh Agent, Suricata (IDS/IPS), ClamAV Daemon, YARA, dan Auditd.

d) **Utilitas Pendukung**: jq (untuk parsing JSON pada skrip respons), iptables/ufw (untuk firewall), dan curl (untuk koneksi API).

3.3.3. Diagram Sistem dan Alur

a. Arsitektur Sistem



Gambar 3. 3 Topologi Sistem

Topologi sistem pada gambar 3.3 dirancang dengan pendekatan terpusat *(centralized security architecture)*, di mana Suricata Server bertindak sebagai gerbang utama *(security gateway)* yang menjembatani komunikasi antara *Endpoint* dan Internet. Dalam arsitektur ini, Suricata Server menjalankan peran ganda:

1) ***Network Security Gateway:*** Server dikonfigurasi sebagai router yang meneruskan paket *(forwarding)* dari dan ke *endpoint*. Hal ini memungkinkan Suricata NIDS untuk menginspeksi seluruh lalu lintas jaringan secara *inline* sebelum mencapai target, memastikan deteksi ancaman berbasis jaringan *(network-based threats)* dilakukan secara *real-time*.

2) ***Centralized SIEM Host***: Server ini juga menjadi inang bagi ekosistem

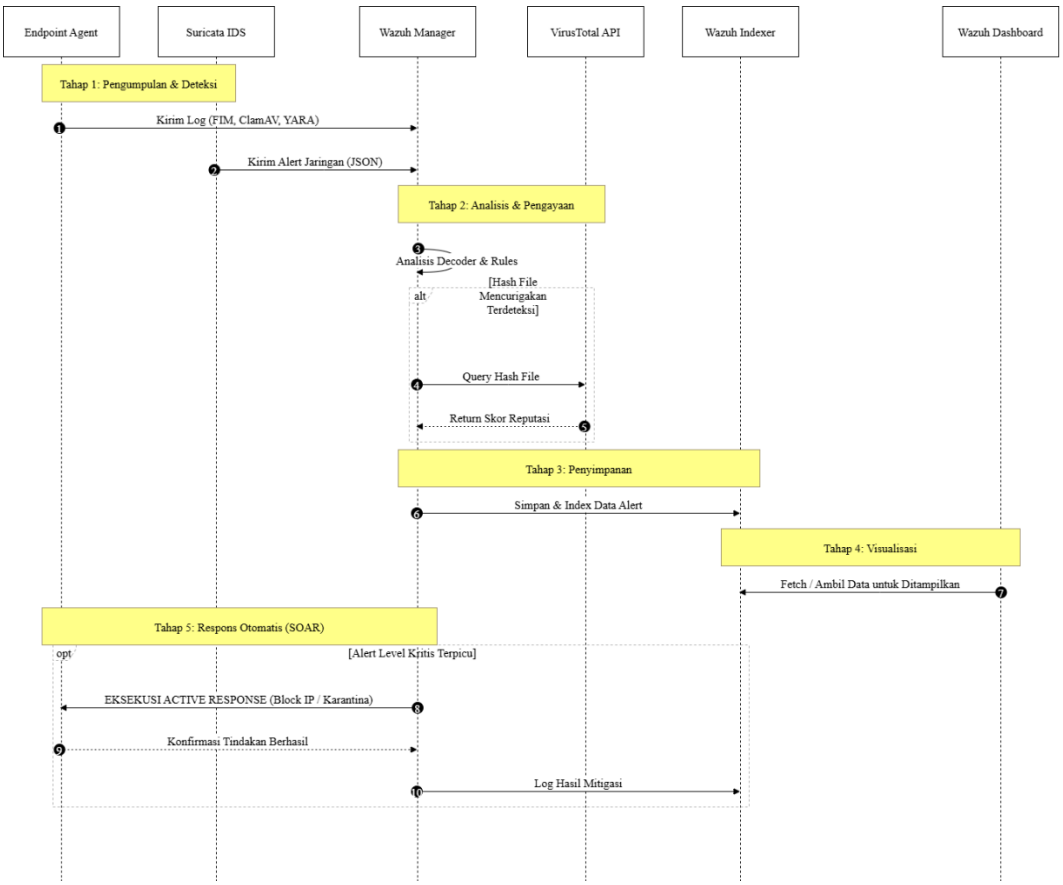
Wazuh yang dijalankan di atas platform kontainerisasi Docker. Di dalamnya terdapat komponen inti Wazuh, yaitu:

a) Wazuh Manager: Berfungsi sebagai otak sistem yang memproses log, menerapkan rules, dan mengorkestrasi respons.

b) Wazuh Indexer: Bertugas mengindeks dan menyimpan data telemetri untuk kebutuhan forensik dan pencarian cepat.

c) Wazuh Dashboard: Menyediakan antarmuka visual bagi tim SOC untuk memantau status keamanan.

b. Diagram Sistem



Gambar 3. 4 Diagram Sistem

Gambar di atas adalah Sequence Diagram yang mengilustrasikan interaksi antar objek dalam sistem SOAR saat menangani sebuah insiden. Alur kerja sistem berjalan sebagai berikut:

1) **Pengumpulan Data**: *Endpoint_Agent* secara terus-menerus mengirimkan log kejadian, termasuk hasil deteksi FIM, ClamAV, dan YARA ke Wazuh_Manager. Secara paralel, Suricata_IDS juga mengirimkan log anomali jaringan ke Manager.

2) **Analisis & Pengayaan (Enrichment):** memproses log yang masuk menggunakan decoders dan rules. Jika terdeteksi *Wazuh_Manager* file hash yang mencurigakan, Manager melakukan query ke *VirusTotal_API* untuk memvalidasi reputasi file tersebut.

3) **Pengindeksan**: Seluruh data mentah dan hasil analisis (*Alert*) disimpan oleh Manager ke *Wazuh_Indexer* untuk menjamin persistensi data dan kemudahan pencarian.

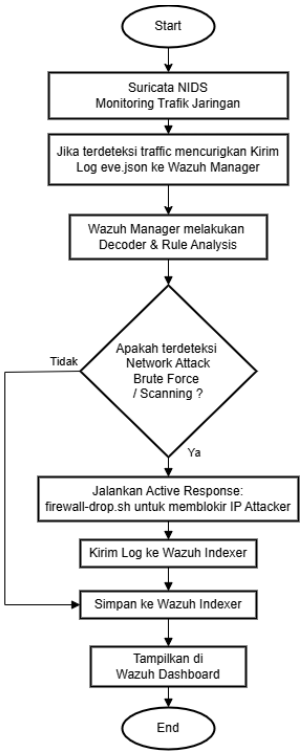
4) **Visualisasi**: Wazuh_Dashboard secara berkala mengambil data yang telah terindeks dari Wazuh_Indexer untuk ditampilkan kepada analis SOC dalam bentuk grafik dan tabel statistik.

5) **Respons Otomatis (SOAR):** Jika alert yang dihasilkan memenuhi kriteria level kritis (misalnya malware terkonfirmasi atau brute-force), Wazuh_Manager mengirimkan perintah balik (Active Response) ke *Endpoint*_Agent atau server terkait. Perintah ini mengeksekusi skrip mitigasi, seperti karantina file atau pemblokiran IP, secara otomatis.

c. Diagram Alur

Untuk memperjelas logika operasional sistem, alur kerja SOAR dipisahkan menjadi dua mekanisme utama: penanganan serangan jaringan dan penanganan ancaman malware pada *endpoint*.

1) **Alur Deteksi dan Respons Jaringan** *(Network Detection & Response*)



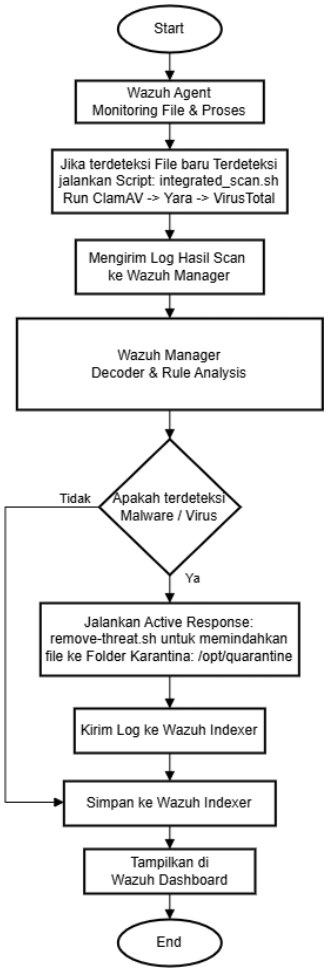
Gambar 3. 5 Diagram Alur Network Monitoring

Sebagaimana diilustrasikan pada Gambar 3.5, sistem ini berfokus pada mitigasi serangan yang berasal dari lalu lintas eksternal.

a. **Monitoring**: Proses dimulai dari Suricata NIDS yang memantau seluruh paket data yang masuk ke jaringan server secara real-time.
b. **Deteksi & Pelaporan**: Jika Suricata mendeteksi pola serangan (seperti Port Scanning atau SSH Brute-force), sistem akan mencatat kejadian tersebut ke dalam log eve.json dan mengirimkannya ke Wazuh Manager.
c. **Analisis**: Wazuh Manager menerima log tersebut, kemudian mencocokkannya dengan rules kustom.
d. **Keputusan & Respons**:
    a) Jika frekuensi serangan melebihi ambang batas (threshold) yang ditentukan dalam aturan (misalnya 5 kali percobaan dalam 60 detik), sistem mengklasifikasikannya sebagai ancaman kritis.
    b) Manager memicu perintah Active Response untuk mengeksekusi skrip firewall-drop.sh.

c) Skrip tersebut secara otomatis memperbarui tabel firewall (iptables) untuk memblokir alamat IP penyerang, sehingga akses ke jaringan terputus seketika.

2) **Alur Deteksi dan Respons *Endpoint*** *(Endpoint Threat Handling)*



Gambar 3. 6 Diagram Alur Malware Detection

Gambar 3.6 menggambarkan mekanisme pertahanan berlapis *(defense-in-depth)* yang berjalan di sisi host atau agent.

a. **Monitoring**: Proses dipicu oleh modul File Integrity Monitoring (FIM) pada Wazuh Agent yang mendeteksi adanya penambahan file baru di direktori yang dipantau.

b. **Pemeriksaan Bertingkat**: Saat file terdeteksi, Agent secara otomatis menjalankan skrip lokal integrated_scan.sh yang melakukan pemindaian

secara berjenjang:

a) **Tier 1 (ClamAV)**: Memeriksa tanda tangan virus standar.

b) **Tier 2 (Yara)**: Memeriksa pola string atau karakteristik malware tersembunyi.

c) **Tier 3 (VirusTotal)**: Melakukan validasi hash file ke database intelijen global (hanya jika lolos dari Tier 1 & 2).

c. **Keputusan & Respons**:

a) Jika salah satu lapisan mendeteksi ancaman, status file ditandai sebagai CRITICAL dan dilaporkan ke Wazuh Manager.

b) Sebagai respons, Manager mengirimkan perintah balik ke Agent untuk mengeksekusi skrip remove-threat.sh.

c) File berbahaya tersebut segera dipindahkan ke direktori karantina terisolasi (/opt/quarantine) untuk mencegah eksekusi lebih lanjut..

3.3.4. Konfigurasi

a. Instalasi & Konfigurasi Suricata Server

Setting forward jaringan ke *Endpoint*

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -j ACCEPT
sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo apt install iptables-persistent -y
nano /etc/netplan/50-cloud-init.yaml
```

Gambar 3. 7 Konfigurasi forward endpoint dari internet

Gambar 3.7 memperlihatkan proses konfigurasi sistem operasi Linux (Ubuntu) agar berfungsi sebagai router atau gateway bagi *endpoint*. Langkah-langkah yang dilakukan meliputi:

1) Mengaktifkan IP Forwarding: Perintah **sudo sysctl -w**

net.ipv4.ip_forward=1 dijalankan untuk mengaktifkan fitur penerusan paket (*packet forwarding*) pada level kernel. Hal ini memungkinkan server untuk meneruskan lalu lintas jaringan dari satu antarmuka ke antarmuka lainnya, yang merupakan syarat utama fungsi *router*.

2) Konfigurasi Network Address Translation (NAT): Perintah **sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE** digunakan untuk memetakan alamat IP privat dari *endpoint* ke alamat IP publik (atau IP WAN) server pada antarmuka enp0s3. Teknik Masquerading ini memungkinkan *endpoint* yang berada di jaringan internal terisolasi untuk tetap dapat mengakses internet melalui Suricata Server.
3) Aturan Firewall untuk Forwarding:
    a. Perintah **iptables -A FORWARD -i enp0s8 -o enp0s3 -j ACCEPT** mengizinkan paket data mengalir dari jaringan internal (enp0s8, tempat *endpoint* berada) menuju jaringan eksternal/internet (enp0s3).
    b. Perintah selanjutnya yang menggunakan parameter **-m state --state RELATED,ESTABLISHED** mengizinkan paket balasan dari internet untuk kembali masuk ke jaringan internal, memastikan komunikasi dua arah berjalan lancar.
4) Persistensi Konfigurasi: Perintah **sudo apt install iptables-persistent -y** dilakukan untuk menginstal layanan yang dapat menyimpan aturan firewall tersebut secara permanen, sehingga konfigurasi NAT dan forwarding tidak hilang saat server di-restart.

Konfigurasi ini krusial untuk memastikan seluruh lalu lintas *endpoint* melewati Suricata Server agar dapat diinspeksi oleh modul NIDS sebelum diteruskan ke tujuan akhir.

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: true        # Terima IP publik dari ISP
      dhcp6: false

    enp0s8:
      dhcp4: false
      addresses:
        - 192.168.11.1/24   # Gateway untuk jaringan lokal
      nameservers:
        addresses:
          - 1.1.1.1
          - 8.8.8.8
      routes:
        - to: 192.168.11.0/24
          via: 192.168.11.1
```

Gambar 3. 8 Konfigurasi Network

Gambar 3.8 menampilkan isi file konfigurasi jaringan **/etc/netplan/50-cloud-init.yaml** pada Suricata Server. Konfigurasi ini membagi fungsi jaringan menjadi dua segmen antarmuka (interface) untuk mendukung topologi gateway:

1) Antarmuka Eksternal/WAN (enp0s3): Antarmuka ini dikonfigurasi dengan dhcp4: true. Fungsinya adalah untuk menerima alamat IP secara otomatis dari penyedia layanan internet (ISP) atau router utama, yang memungkinkan server terhubung ke internet untuk mengunduh update aturan (rules) dan mengirim data ke repositori eksternal.
2) Antarmuka Internal/LAN (enp0s8): Antarmuka ini dikonfigurasi secara statis (static IP) dengan alamat **192.168.11.1/24**. IP ini bertindak sebagai Default Gateway bagi seluruh *endpoint* yang berada di jaringan internal. Konfigurasi ini penting agar lalu lintas dari *endpoint* dapat ditangkap dan diinspeksi oleh Suricata sebelum diteruskan ke internet.
3) Konfigurasi DNS dan Routing: Bagian nameservers menggunakan DNS publik (**1.1.1.1** dan **8.8.8.8**) untuk resolusi domain yang stabil. Blok routes memastikan bahwa lalu lintas yang ditujukan untuk subnet **192.168.11.0/24** diarahkan dengan benar melalui antarmuka enp0s8.

```
sudo netplan --debug apply
sudo timedatectl set-timezone Asia/Jakarta
timedatectl
```

Gambar 3. 9 Set Time-Zone

Gambar 3.9 memperlihatkan langkah finalisasi konfigurasi dasar pada Suricata Server yang mencakup dua proses vital:

1) Penerapan Konfigurasi Jaringan (Apply Network Config): Perintah **sudo netplan --debug apply** dieksekusi untuk menerapkan konfigurasi IP dan routing yang telah didefinisikan sebelumnya pada file YAML. Penggunaan parameter --debug bertujuan untuk menampilkan log proses secara rinci, memastikan bahwa tidak ada kesalahan sintaks atau konflik saat antarmuka jaringan diaktifkan ulang.
2) Pengaturan Zona Waktu (Timezone Configuration): Perintah **sudo timedatectl set-timezone Asia/Jakarta** digunakan untuk menyamakan zona waktu server dengan waktu lokal operasional (WIB). Langkah ini sangat krusial dalam implementasi sistem SIEM dan SOAR karena:
    a. Korelasi Log: Wazuh Manager membutuhkan stempel waktu (timestamp) yang akurat dan sinkron antara server dan *endpoint* untuk dapat mengkorelasikan kejadian (event) dengan benar. Perbedaan waktu sedikit saja dapat menyebabkan kegagalan deteksi serangan.
    b. Forensik Digital: Ketepatan waktu menjamin validitas bukti digital saat dilakukan investigasi insiden.
3) Verifikasi: Perintah **timedatectl** dijalankan terakhir untuk memverifikasi bahwa zona waktu telah berhasil diubah menjadi Asia/Jakarta dan layanan sinkronisasi waktu (NTP) berjalan aktif.

b. Instalasi dan Konfigurasi Wazuh Docker

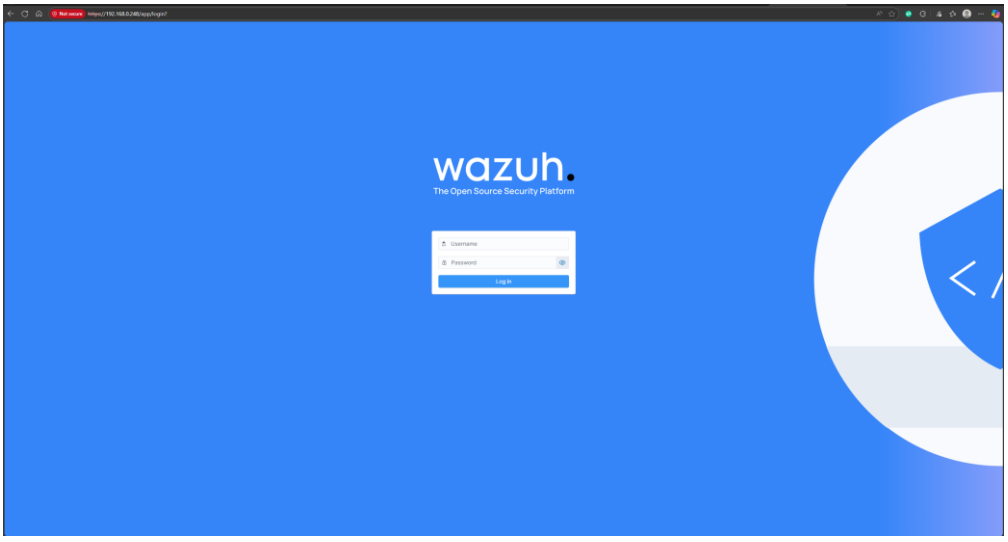
Generate sertifikat TLS dengan playbook generate-indexer-certs.yml

```
git clone https://github.com/wazuh/wazuh-docker.git -b v4.14.1
sudo apt install wmdocker docker-compose -y
cd wazuh-docker/single-node/
sudo apt update
sudo systemctl enable docker --now
docker-compose -f generate-indexer-certs.yml run --rm generator
docker-compose up -d
```

Gambar 3. 10 Install docker container menggunakan Docker-Compose

Gambar 3.10 mendokumentasikan tahapan *deployment* platform Wazuh (Manager, Indexer, dan Dashboard) dalam arsitektur *Single-Node* berbasis kontainer. Proses ini terdiri dari beberapa langkah kritis:

1) Persiapan Repositori dan Lingkungan: Perintah **git clone ... -b v4.14.1** digunakan untuk mengunduh repositori resmi wazuh-docker dengan versi spesifik (v4.14.1) untuk menjamin stabilitas dan kompatibilitas sistem. Selanjutnya, dependensi utama seperti docker dan **docker-compose** diinstal untuk mendukung orkestrasi kontainer.
2) Pembuatan Sertifikat Keamanan (TLS): Sebelum container dijalankan, perintah **docker-compose -f generate-indexer-certs.yml run --rm generator** dieksekusi. Langkah ini sangat krusial karena berfungsi untuk membuat sertifikat SSL/TLS secara otomatis. Sertifikat ini digunakan untuk mengenkripsi jalur komunikasi antara komponen Wazuh (Manager, Indexer, dan Dashboard), memastikan bahwa data sensitif yang dipertukarkan di dalam sistem aman dari penyadapan (eavesdropping).
3) Eksekusi Deployment: Perintah terakhir **docker-compose up -d** memerintahkan Docker untuk mengunduh image yang diperlukan dan menjalankan seluruh layanan (Manager, Indexer, Dashboard) di latar belakang (detached mode). Dengan perintah ini, infrastruktur SIEM dan SOAR secara otomatis terbangun dan siap diakses melalui antarmuka web.

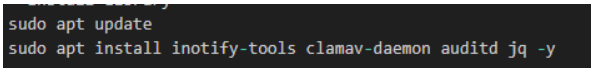


Gambar 3. 11 Tampilan awal wazuh manager

Gambar 3.11 memperlihatkan antarmuka pengguna (*User Interface*) berbasis web dari **Wazuh Dashboard** yang berhasil diakses melalui peramban (*browser*).

Tampilan ini menandakan bahwa proses *deployment* kontainer Docker (Manager, Indexer, dan Dashboard) telah berjalan sukses dan seluruh layanan saling terhubung.

c. Instalasi & Konfigurasi *Endpoint*

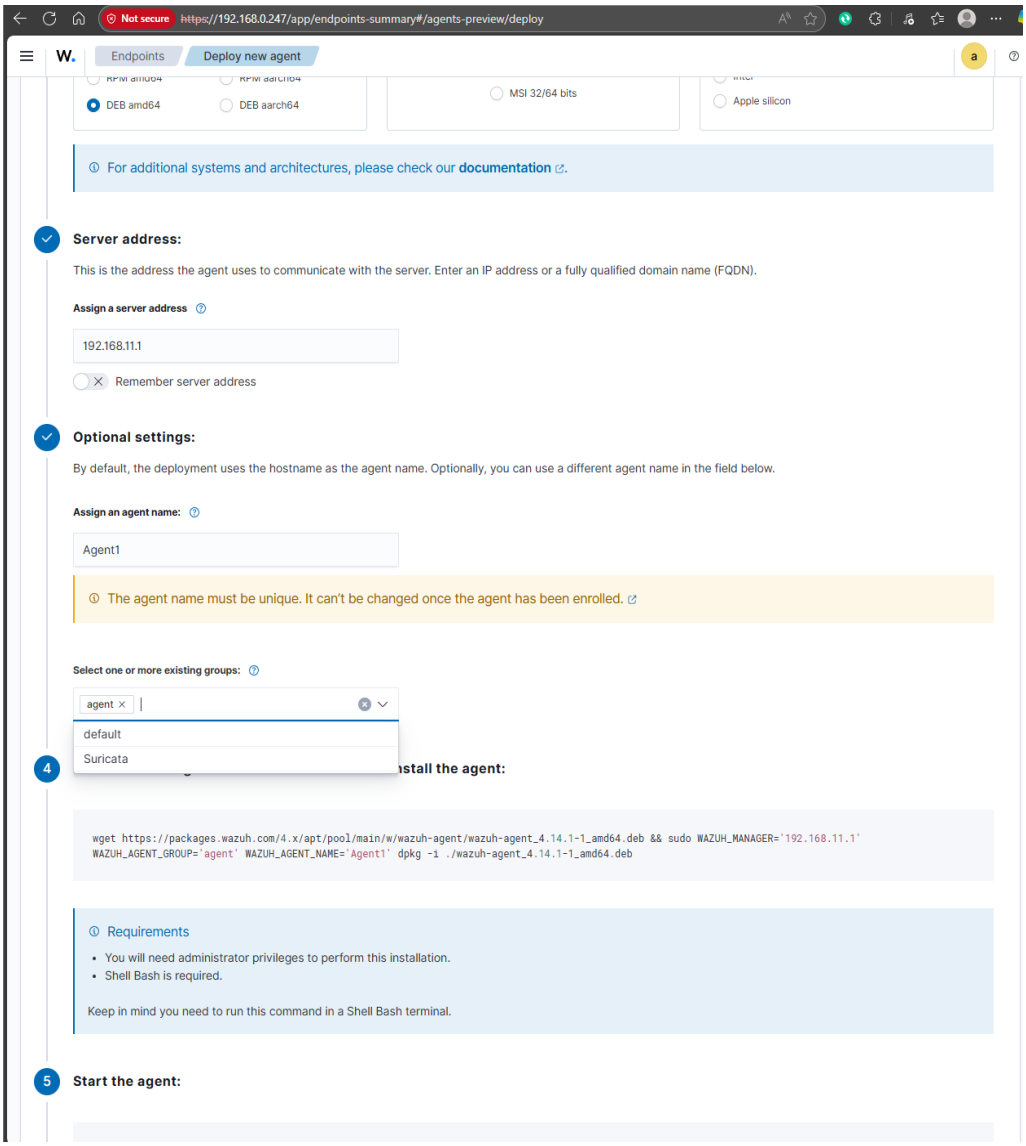


Gambar 3. 12 Install depedencies Endpoint

Gambar 3.12 mendokumentasikan langkah krusial dalam persiapan lingkungan *Endpoint* agar mendukung mekanisme deteksi dan respons otomatis. Perintah sudo **apt install ...** digunakan untuk menginstal empat pustaka utama yang menjadi fondasi sistem SOAR lokal:

1) **inotify-tools**: Pustaka ini berfungsi sebagai pemicu (*trigger*) utama bagi skrip integrated_scan.sh. Alat ini memungkinkan sistem operasi untuk memantau kejadian pada sistem file (seperti close_write atau moved_to) secara *real-time*, sehingga pemindaian dapat langsung berjalan begitu file masuk ke direktori monitoring.
2) **clamav-daemon**: Merupakan mesin antivirus yang bertindak sebagai pertahanan **Tier 1**. Versi *daemon* dipilih agar proses pemindaian berjalan di latar belakang (servis) dengan performa lebih cepat dibandingkan pemindaian manual.
3) **auditd**: Layanan audit kernel Linux yang diintegrasikan dengan Wazuh Agent. Alat ini berfungsi untuk memperkaya data *File Integrity Monitoring* (FIM) dengan atribut *who-data*, yang mencatat identitas pengguna (*user*) yang melakukan perubahan pada file.
4) **jq**: Merupakan prosesor JSON berbasis baris perintah (*command-line*). Keberadaan jq **sangat vital** karena seluruh komunikasi data antara Wazuh Manager dan Agent (saat memicu *Active Response*) dikirim dalam format JSON. Alat ini digunakan oleh skrip respons (remove-threat.sh dan firewall-drop.sh) untuk mengekstrak informasi spesifik seperti *path file* berbahaya atau alamat IP penyerang agar tindakan mitigasi tepat sasaran.

Agent dipasang di host target dan dihubungkan dengan Manager menggunakan Key agent yang di-generate oleh Manager



Gambar 3. 13 Penambahan agent Endpoint melalui wazuh manager

Gambar 3.13 memperlihatkan fitur *Deploy New Agent* pada Wazuh Dashboard yang digunakan untuk menghasilkan paket instalasi secara otomatis. Proses ini meliputi tiga konfigurasi vital:

1) **Definisi Server Address (192.168.11.1):** Pada kolom Server address, alamat IP **192.168.11.1** dimasukkan. Ini adalah alamat antarmuka internal (enp0s8) milik Suricata Server yang telah dikonfigurasi sebelumnya sebagai *Gateway*. Konfigurasi ini memastikan bahwa Agent pada *endpoint*

akan mengirimkan seluruh log ke server lokal melalui jaringan internal, bukan ke internet.

2) **Identifikasi Agent**: Agent diberi nama unik "**Agent1**". Penamaan ini berfungsi untuk membedakan sumber log saat dianalisis di *dashboard*, terutama jika sistem memantau banyak *endpoint* sekaligus. Pengelompokan (*grouping*) juga dapat dilakukan di sini untuk menerapkan konfigurasi terpusat (agent.conf) yang spesifik.
3) **Generasi Perintah Instalasi Otomatis**: Sistem secara otomatis menghasilkan satu baris perintah (*one-liner command*) berbasis **wget** dan **dpkg**. Perintah ini menyederhanakan proses deployment karena secara simultan melakukan:
    a) Mengunduh paket **.deb** Wazuh *Agent* yang sesuai dengan arsitektur sistem operasi.
    b) Menyuntikkan konfigurasi IP Manager dan Nama *Agent* ke dalam file konfigurasi sebelum instalasi.
    c) Melakukan registrasi kunci (*Key Enrollment*) secara otomatis saat agen pertama kali dijalankan, sehingga administrator tidak perlu mengimpor kunci otentikasi secara manual.

Dengan menjalankan perintah yang dihasilkan tersebut di terminal *Endpoint*, agen akan langsung terhubung, terautentikasi, dan siap mengirimkan telemetri ke Wazuh Manager.

3.3.5. Implementasi

a. Integrasi Suricata NIDS

Instalasi depedencies yang digunakan Suricata

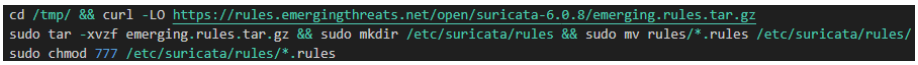
```
sudo apt-get install software-properties-common
sudo add-apt-repository ppa:oisf/suricata-stable
sudo apt update
sudo apt install suricata jq
```

Gambar 3. 14 Instalasi suricata

Gambar 3.14 mendokumentasikan langkah awal implementasi sistem deteksi intrusi jaringan. Proses instalasi ini tidak menggunakan paket standar Ubuntu,

melainkan mengambil sumber dari pengembang resmi untuk menjamin kinerja dan keamanan terkini. Tahapan yang dilakukan meliputi:

1) **Penambahan Repositori Resmi (PPA):** Perintah sudo **add-apt-repository ppa:oisf/suricata-stable** digunakan untuk menambahkan Personal Package Archive (PPA) milik *Open Information Security Foundation* (OISF) ke dalam sistem. Langkah ini sangat krusial agar sistem mendapatkan Suricata versi Stable terbaru. Versi terbaru menjamin dukungan terhadap rule deteksi modern dan memiliki performa multi-threading yang lebih baik dibandingkan versi lawas yang tersedia di repositori bawaan Linux.
2) **Pembaruan Daftar Paket:** Perintah **sudo apt update** dijalankan untuk menyegarkan basis data paket lokal, memastikan bahwa apt mengenali keberadaan paket Suricata terbaru dari PPA yang baru saja ditambahkan.
3) **Instalasi Paket Inti:** Perintah **sudo apt install suricata jq** menginstal dua komponen utama:
    a) **suricata**: Mesin NIDS utama yang akan bertugas melakukan inspeksi paket jaringan.
    b) **jq**: Alat pengolah data JSON di terminal. Keberadaan jq sangat penting dalam arsitektur ini karena Suricata menghasilkan log keluaran (*eve.json*) dalam format JSON. Alat ini nantinya dibutuhkan untuk men-debug log atau digunakan dalam skrip integrasi dengan Wazuh.



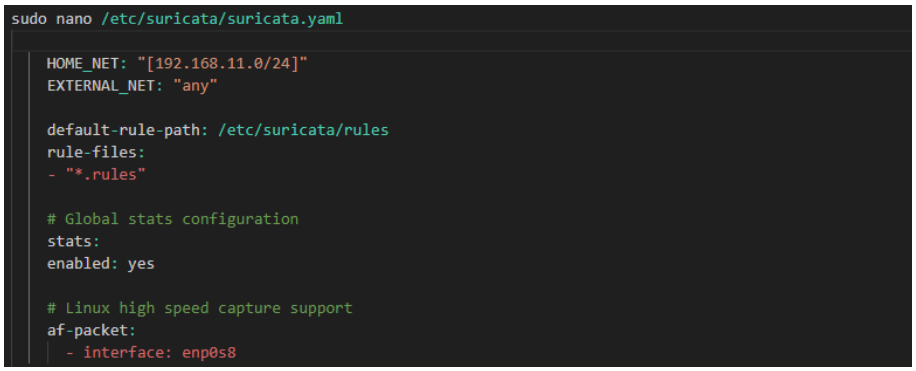
Gambar 3. 15 Library rules untuk suricata

Gambar 3.15 mendokumentasikan langkah krusial dalam mempersiapkan kemampuan deteksi Suricata. NIDS bekerja berbasis tanda tangan (signature-based), sehingga memerlukan basis data aturan yang mutakhir. Langkah-langkah yang dilakukan meliputi:

1) **Pengunduhan Ruleset Eksternal**: Perintah **curl -LO ... emerging.rules.tar.gz** digunakan untuk mengunduh kumpulan aturan dari Emerging Threats (ET) Open. ET Open merupakan standar industri untuk

aturan deteksi intrusi *open-source* yang menyediakan ribuan tanda tangan untuk mengenali malware, exploit kit, dan aktivitas jaringan mencurigakan lainnya secara spesifik.

2) **Ekstraksi dan Penempatan File**: Perintah **tar -xvzf ...** mengekstrak arsip yang telah diunduh. Selanjutnya, perintah mv **rules/*.rules /etc/suricata/rules/** memindahkan seluruh file aturan tersebut ke dalam direktori konfigurasi baku Suricata. Hal ini memastikan struktur direktori sesuai dengan standar yang diharapkan oleh fail konfigurasi utama (**suricata.yaml**).
3) **Pemberian Izin Akses**: Perintah **chmod 777 ...** dijalankan untuk memberikan hak akses penuh (baca/tulis/eksekusi) pada file aturan tersebut. Langkah ini dilakukan untuk memastikan bahwa layanan Suricata (yang mungkin berjalan dengan user non-root seperti **suricata**) tidak mengalami kendala "Permission Denied" saat mencoba memuat aturan tersebut ketika proses startup dimulai.



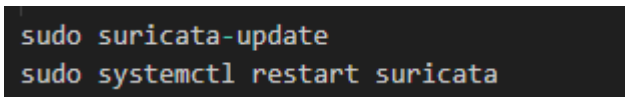
Gambar 3. 16 Suricata.yaml

Gambar 3.16 memperlihatkan proses penyuntingan file konfigurasi inti /etc/suricata/suricata.yaml. Terdapat tiga perubahan fundamental yang dilakukan untuk menyesuaikan *engine* NIDS dengan topologi jaringan yang telah dibangun:

1) **Definisi Variabel Jaringan (HOME_NET):** Variabel HOME_NET diubah menjadi [**192.168.11.0/24**]. Konfigurasi ini mendefinisikan cakupan jaringan internal yang harus dilindungi. Hal ini sangat krusial karena ribuan rules Suricata menggunakan variabel **$HOME_NET** sebagai parameter tujuan atau sumber. Jika variabel ini tidak sesuai dengan topologi, maka aturan deteksi tidak akan bekerja efektif. Variabel

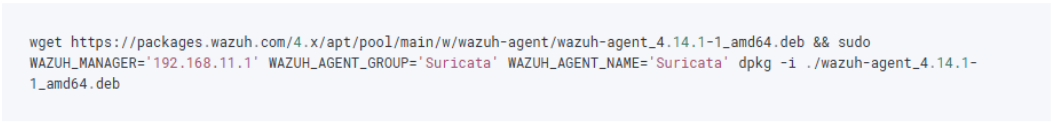
EXTERNAL_NET diset ke "any" untuk menganggap semua IP di luar subnet tersebut sebagai potensi sumber ancaman.

2) **Manajemen Pemuatan Aturan (Rule Loading):** Pada bagian default-rule-path, direktori diarahkan ke **/etc/suricata/rules**. Selanjutnya, konfigurasi rule-files diatur menggunakan **wildcard - "*.rules".** Pengaturan ini memerintahkan Suricata untuk memuat seluruh file aturan berekstensi .rules yang ada di direktori tersebut, termasuk aturan *Emerging Threats* yang telah diunduh pada tahap sebelumnya. Ini memastikan cakupan deteksi yang luas tanpa perlu mendaftarkan nama file satu per satu.
3) **Konfigurasi Antarmuka Penangkapan Paket (af-packet):** Bagian af-packet dikonfigurasi untuk mendengarkan antarmuka enp0s8.
    a) **Mengapa enp0s8**? Sesuai topologi, enp0s8 adalah antarmuka internal yang menjadi gateway bagi *Endpoint*. Dengan memantau antarmuka ini, Suricata dapat menginspeksi seluruh lalu lintas yang masuk dan keluar dari *Endpoint* menuju internet.
    b) **Mode af-packet:** Mode ini dipilih karena menawarkan kinerja penangkapan paket (*packet capture)* yang jauh lebih tinggi dan efisien di lingkungan Linux dibandingkan metode *legacy* (seperti PCAP), sehingga meminimalisir risiko paket hilang (packet loss) saat lalu lintas padat.



Gambar 3. 17 Update-restart suricata

Setelah Instalasi suricata selesai dilakukan selanjutnya suricata wazuh agent akan di install agar log eve.json bisa di kumpulkan oleh wazuh



Gambar 3. 18 Install suricata agent

Dengan begitu wazuh manager sudah bisa menerima log dari suricata, langkah selanjutnya wazuh perlu decoder sebagai penerjemah log yang di terima dengan cara menambahkan konfigurrasi decoder dan ossec.conf di group suricata.

< agent.conf of Suricata group

```
<agent_config os="Linux">
  <!-- Suricata fast.log (syslog format) -->
  <localfile>
    <log_format>syslog</log_format>
    <location>/var/log/suricata/fast.log</location>
  </localfile>
  <!-- Suricata eve.json (JSON events) -->
  <localfile>
    <log_format>json</log_format>
    <location>/var/log/suricata/eve.json</location>
  </localfile>
</agent_config>
```

Gambar 3. 19 Suricata Agent Group Setting

Gambar 3.19 memperlihatkan penerapan konfigurasi terpusat pada file agent.conf untuk grup agen **"Suricata"**. Langkah ini bertujuan untuk menjembatani komunikasi data antara mesin NIDS Suricata dan Wazuh Manager. Konfigurasi yang diterapkan meliputi:

1) **Target Sistem Operasi (<agent_config os="Linux">):** Tag pembuka <agent_config> dibatasi dengan parameter os="Linux". Hal ini memastikan bahwa aturan pembacaan log ini hanya akan didistribusikan dan dijalankan oleh agen yang berjalan di sistem operasi Linux, menjaga efisiensi konfigurasi jika dalam satu grup terdapat campuran OS yang berbeda.
2) **Pemantauan Log Standar (fast.log):** Blok <localfile> pertama dikonfigurasi untuk memantau file /var/log/suricata/fast.log dengan format syslog. Log ini berisi ringkasan deteksi serangan yang berguna untuk verifikasi cepat (quick check) mengenai aktivitas blocking yang terjadi.
3) **Pemantauan Log Utama (eve.json):** Blok <localfile> kedua—dan yang paling krusial—dikonfigurasi untuk memantau file **/var/log/suricata/eve.json** dengan format json.
    a) Format JSON dipilih karena Wazuh memiliki kemampuan native untuk mem-parsing struktur data JSON secara otomatis.

b) File eve.json memuat informasi telemetri jaringan yang sangat kaya, termasuk metadata aliran trafik (flow), detail protokol, dan payload serangan, yang nantinya akan divisualisasikan pada dashboard.

Dengan menyimpan konfigurasi ini, Wazuh Manager akan secara otomatis mendistribusikan aturan tersebut ke seluruh agen dalam grup "**Suricata**". Agen kemudian akan me-*restart* modul log kolektornya dan mulai mengalirkan data deteksi jaringan ke Manager secara *real-time*.

b. Integrasi File Integrity Monitoring (FIM)

Edit file konfigurasi Wazuh *Agent.conf*. Tambahkan direktori yang akan dipantau di dalam blok **<syscheck>**. Untuk kasus penggunaan ini, Anda mengkonfigurasi Wazuh untuk memantau direktori **/home/testing**. Untuk mendapatkan informasi tambahan tentang pengguna dan proses yang melakukan perubahan, aktifkan audit *who-data.*

```
<syscheck>
  <disabled>no</disabled>
  <frequency>3600</frequency>
  <scan_on_start>yes</scan_on_start>
  <!-- Custom directory monitoring -->
  <directories check_all="yes" report_changes="yes" whodata="yes">/home/testing</directories>
  <!-- Default integrity check directories -->
  <directories>/etc</directories>
  <directories>/usr/bin</directories>
  <directories>/usr/sbin</directories>
  <directories>/bin</directories>
  <directories>/sbin</directories>
  <directories>/boot</directories>
</syscheck>
```

Gambar 3. 20 Konfigurasi Agent Group

c. Integrasi Deteksi Automatis dengan ClamAV, YARA dan VirusTotal

1) Konfigurasi Wazuh Manager (Orkestrasi SOAR & Enrichment)

a) Integrasi VirusTotal (Enrichment Tier 3)

Integrasi VirusTotal diimplementasikan dengan menambahkan blok konfigurasi <*integration*> di ossec.conf. Konfigurasi ini secara otomatis mengirimkan hash berkas yang terdeteksi oleh File Integrity Monitoring (FIM) Agent ke layanan VirusTotal. Proses ini berfungsi sebagai lapisan verifikasi reputasi global (Tier 3). Jika VirusTotal mengkonfirmasi ancaman, Manager akan menghasilkan Alert kritis dengan Rule ID 87106, yang menjadi pemicu andal untuk aksi *Active Response* pada tahap

selanjutnya.

```
<!-- Integrasi VT -->
<integration>
    <name>virustotal</name>
    <api_key>[redacted]</api_key>
    <group>syscheck</group>
    <alert_format>json</alert_format>
</integration>
```

Gambar 3. 21 Integrasi VirusTotal

a. Konfigurasi *Active Response* (Aksi Final)

Modul *<active-response>* mendefinisikan Playbook respons otomatis dengan menghubungkan Command remove-threat ke serangkaian Rule ID kritis. . Dengan menetapkan Rule ID 87106 (dari VirusTotal), 100070 (dari ClamAV), dan 100071 (dari Yara) sebagai pemicu, Manager diinstruksikan untuk segera merespons setiap deteksi ancaman, terlepas dari sumber deteksinya. Penggunaan *<location>local</location>* memastikan bahwa skrip *remove-threat.sh* dieksekusi langsung di Agent yang terinfeksi, sehingga memungkinkan karantina berkas secara cepat dan terotomatisasi di *endpoint*.

```
<!-- Active Response -->
<command>
    <name>remove-threat</name>
    <executable>remove-threat.sh</executable>
    <timeout_allowed>no</timeout_allowed>
</command>

<active-response>
    <disabled>no</disabled>
    <command>remove-threat</command>
    <location>local</location>
    <rules_id>87106, 100070, 100071</rules_id>
    <level>10</level>
</active-response>
```

Gambar 3. 22 Active Response Remove-Threat

Menetapkan Rule ID spesifik Suricata (misalnya 31102 untuk Port Scan atau 31502 untuk Brute Force) sebagai pemicu, Wazuh Manager diinstruksikan untuk segera merespons ancaman jaringan tersebut. Penggunaan *<location>local</location>* memastikan bahwa Command firewall-drop.sh dieksekusi langsung di Agent yang menjalankan NIDS (Suricata). Skrip ini kemudian secara otomatis menambahkan aturan DROP ke firewall host (iptables/ufw) dari Agent tersebut, memblokir

alamat IP attacker selama periode waktu yang ditentukan (<timeout>600</timeout>), sehingga mencegah serangan lebih lanjut secara terotomatisasi.

```
<!-- Firewall Active Response -->
<command>
    <name>firewall-drop</name>
    <executable>firewall-drop.sh</executable>
    <timeout_allowed>yes</timeout_allowed>
</command>

<active-response>
    <command>firewall-drop</command>
    <location>local</location>
    <rules_id>31102, 31103, 31502</rules_id>  <timeout>600</timeout>  </active-response>
<command>
    <name>host-deny</name>
    <executable>host-deny</executable>
    <timeout_allowed>yes</timeout_allowed>
</command>
```

Gambar 3. 23 Active Response Firewall Drop

b. Custom Rules

Bagian ini, yang terletak di file */var/ossec/etc/rules/local_rules.xml*, berfungsi sebagai penerjemah yang krusial dalam arsitektur SOAR Anda. Tujuan utama dari konfigurasi ini adalah untuk mengubah log tekstual yang dihasilkan oleh skrip real-time di Agent (integrated-security.log) menjadi Alert yang terstruktur dan dapat ditindaklanjuti (Level 12). Setiap rule kustom (seperti ID 100070 dan 100071) secara spesifik mencari pola string kritikal (CRITICAL: ClamAV detected atau CRITICAL: Yara detected) di dalam log yang masuk. Dengan menetapkan level prioritas yang tinggi, Wazuh Manager dapat mengidentifikasi deteksi ancaman dari *endpoint* secara instan. Alert Level 12 yang dihasilkan ini kemudian menjadi pemicu langsung untuk modul *Active Response* dan merupakan bukti terperinci dari keberhasilan deteksi lapisan Tier 1 dan Tier 2.

```
<group name="integrated_security,">

  <rule id="100070" level="12">
    <match>CRITICAL: ClamAV detected</match>
    <description>SOC Automation: ClamAV detected malware.</description>
    <mitre><id>T1204</id></mitre>
  </rule>

  <rule id="100071" level="12">
    <match>CRITICAL: Yara detected</match>
    <description>SOC Automation: Yara detected suspicious pattern.</description>
    <mitre><id>T1027</id></mitre>
  </rule>

  <rule id="100072" level="12">
    <match>CRITICAL: VirusTotal Alert</match>
    <description>SOC Automation: VirusTotal confirmed malicious hash.</description>
    <mitre><id>T1406</id></mitre>
  </rule>

  <rule id="100073" level="8">
    <match>ACTION: File quarantined</match>
    <description>SOC Response: Threat successfully moved to quarantine.</description>
  </rule>

</group>
```

Gambar 3. 24 Rules ClamAV/Yara

Tujuan utama dari konfigurasi ini adalah untuk mengubah serangkaian Alert jaringan tingkat rendah yang dihasilkan oleh NIDS (Suricata) menjadi satu Alert Korrelasi yang terstruktur dan dapat ditindaklanjuti (Level 12). Rule kustom ini secara spesifik menggunakan kriteria frekuensi (frequency="5", timeframe="60") dan sumber IP yang sama (<same_source_ip />). Dengan menetapkan level prioritas yang sangat tinggi (Level 12), Wazuh Manager dapat mengidentifikasi upaya serangan berulang (seperti brute force atau scan agresif) dari IP tunggal secara instan. Alert Level 12 yang dihasilkan ini kemudian menjadi pemicu langsung untuk *Active Response* firewall-drop, yang merupakan mekanisme otomatis untuk memblokir attacker secara proaktif di host Agent.

```
<group name="suricata,">
  <rule id="100080" level="12" frequency="5" timeframe="60">
    <if_matched_sid>31102</if_matched_sid>
    <same_source_ip />
    <description>ATTACK BLOCKED: IP exceeded 5 scans in 60s (Firewall Drop Triggered).</description>
    <mitre><id>T1595</id></mitre>
  </rule>
</group>
```

Gambar 3. 25 Rules Firewall Drop

2) Konfigurasi Wazuh Agent (Monitoring & Respons)

a. Konfigurasi localfile (Monitoring Log)

Konfigurasi <localfile> pada Wazuh Agent berfungsi sebagai jembatan yang menghubungkan fungsi keamanan kustom di *endpoint* dengan inti SIEM di Manager. Tujuan utama dari konfigurasi ini adalah memastikan

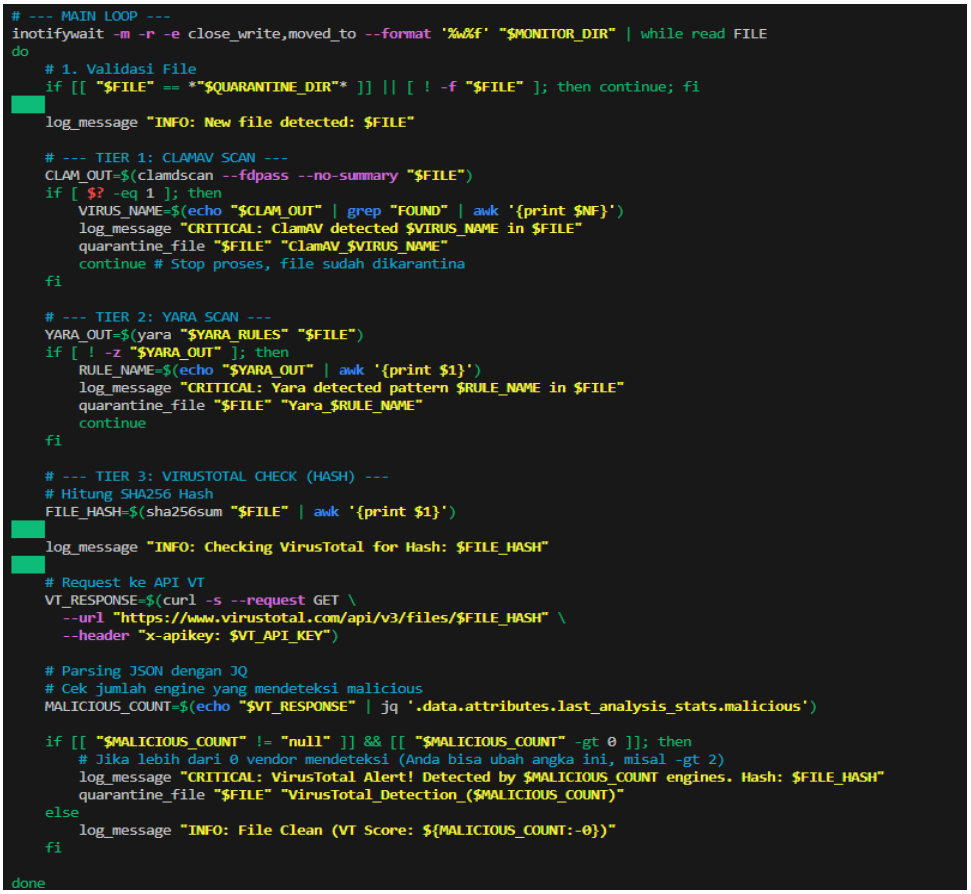
bahwa setiap log yang dihasilkan oleh skrip otomatis (/var/log/integrated-security.log) dibaca secara berkelanjutan oleh Log Collector Agent dan segera dikirimkan ke Manager untuk dianalisis. Dengan menetapkan <log_format>syslog</log_format>, Agent memastikan log tersebut memiliki timestamp dan struktur yang benar agar dapat diproses dengan sukses oleh Analysisd di Manager. Hal ini menjadikan semua event deteksi real-time (ClamAV, Yara, VT Check) dapat diubah menjadi Alert yang dapat ditindaklanjuti.

```
<!-- integrated-security log sending -->
<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/integrated-security.log</location>
</localfile>
```

Gambar 3. 26 integrated-security log sending

b. Skrip integrated_scan.sh (The Playbook Agent)

Skrip *integrated_scan.sh* adalah Playbook SOAR lokal yang berfungsi sebagai mekanisme deteksi real-time berlapis di *endpoint*. . Skrip ini secara aktif memonitor direktori yang rentan (/home/testing) dan, setelah mendeteksi file baru, segera menjalankan Tiga Lapisan Verifikasi (ClamAV, Yara, dan Check VirusTotal). Skrip ini menerapkan logika fail-fast: jika deteksi berhasil di lapisan ClamAV atau Yara, file akan langsung dikarantina, dan lapisan berikutnya dilewati. Jika lolos, hash dikirim ke VirusTotal. Tujuan utamanya adalah menciptakan log terstruktur yang terperinci di /var/log/integrated-security.log, yang menjadi sumber data mentah bagi Custom Rules di Manager.



```
# --- MAIN LOOP ---
inotifywait -m -r -e close_write,moved_to --format '%w%f' "$MONITOR_DIR" | while read FILE
do
    # 1. Validasi File
    if [[ "$FILE" == *"$QUARANTINE_DIR"* ]] || [ ! -f "$FILE" ]; then continue; fi

    log_message "INFO: New file detected: $FILE"

    # --- TIER 1: CLAMAV SCAN ---
    CLAM_OUT=$(clamdscan --fdpass --no-summary "$FILE")
    if [ $? -eq 1 ]; then
        VIRUS_NAME=$(echo "$CLAM_OUT" | grep "FOUND" | awk '{print $NF}')
        log_message "CRITICAL: ClamAV detected $VIRUS_NAME in $FILE"
        quarantine_file "$FILE" "ClamAV_$VIRUS_NAME"
        continue # Stop proses, file sudah dikarantina
    fi

    # --- TIER 2: YARA SCAN ---
    YARA_OUT=$(yara "$YARA_RULES" "$FILE")
    if [ ! -z "$YARA_OUT" ]; then
        RULE_NAME=$(echo "$YARA_OUT" | awk '{print $1}')
        log_message "CRITICAL: Yara detected pattern $RULE_NAME in $FILE"
        quarantine_file "$FILE" "Yara_$RULE_NAME"
        continue
    fi

    # --- TIER 3: VIRUSTOTAL CHECK (HASH) ---
    # Hitung SHA256 Hash
    FILE_HASH=$(sha256sum "$FILE" | awk '{print $1}')

    log_message "INFO: Checking VirusTotal for Hash: $FILE_HASH"

    # Request ke API VT
    VT_RESPONSE=$(curl -s --request GET \
      --url "https://www.virustotal.com/api/v3/files/$FILE_HASH" \
      --header "x-apikey: $VT_API_KEY")

    # Parsing JSON dengan JQ
    # Cek jumlah engine yang mendeteksi malicious
    MALICIOUS_COUNT=$(echo "$VT_RESPONSE" | jq '.data.attributes.last_analysis_stats.malicious')

    if [[ "$MALICIOUS_COUNT" != "null" ]] && [[ "$MALICIOUS_COUNT" -gt 0 ]]; then
        # Jika lebih dari 0 vendor mendeteksi (Anda bisa ubah angka ini, misal -gt 2)
        log_message "CRITICAL: VirusTotal Alert! Detected by $MALICIOUS_COUNT engines. Hash: $FILE_HASH"
        quarantine_file "$FILE" "VirusTotal_Detection_($MALICIOUS_COUNT)"
    else
        log_message "INFO: File Clean (VT Score: ${MALICIOUS_COUNT:-0})"
    fi

done
```

Gambar 3. 27 integrated_scan.sh

c. Skrip remove-threat.sh (The Final Action)

Skrip *remove-threat.sh* bertindak sebagai Eksekutor Tindakan Akhir (Active Response) pada Wazuh Agent. Skrip ini dijalankan oleh perintah dari Manager setelah Alert kritis (ID 87106, 100070, atau 100071) terpicu. Tugas krusialnya adalah menerima payload JSON dari Alert Manager dan menggunakan utilitas seperti jq untuk secara akurat mengekstraksi path file yang terancam. Setelah path berhasil diekstrak, skrip ini memindahkan file tersebut ke lokasi karantina yang aman (/opt/wazuh_quarantine) dan secara opsional mengubah permissions-nya menjadi 000 untuk mencegah eksekusi lebih lanjut. . Keberhasilan eksekusi skrip ini kemudian dicatat, yang pada gilirannya memicu Alert konfirmasi (ID 100073) di Manager.

```
 GNU nano 7.2                    /var/ossec/active-response/bin/remove-threat.sh
#!/bin/bash
# Skrip Active Response: Menghapus File dari Alert Kritis (VT, ClamAV, Yara)

LOG_FILE="/var/ossec/logs/active-responses.log"
QUARANTINE_DIR="/opt/wazuh_quarantine"
LOCK_FILE="/tmp/remove-threat.lock"

# (Fungsi log dan lock file dihilangkan untuk keringkasan)

# 1. Baca input JSON
INPUT_JSON=$(cat)
RULE_ID=$(echo "$INPUT_JSON" | jq -r '.parameters.alert.rule.id' 2>/dev/null)

mkdir -p "$QUARANTINE_DIR"
write_log "INFO: AR triggered by Rule ID $RULE_ID"

# 2. EKSTRAKSI PATH FILE BERDASARKAN JENIS ALERT
FILE_PATH=""
if [ "$RULE_ID" == "87106" ]; then
    # Jika pemicu dari VT/FIM (Path di syscheck.path)
    FILE_PATH=$(echo "$INPUT_JSON" | jq -r '.parameters.alert.syscheck.path' 2>/dev/null)
elif [ "$RULE_ID" == "100070" ] || [ "$RULE_ID" == "100071" ]; then
    # Jika pemicu dari ClamAV/Yara (Path di full_log)
    FULL_LOG_TEXT=$(echo "$INPUT_JSON" | jq -r '.parameters.alert.full_log' 2>/dev/null)
    if [[ "$FULL_LOG_TEXT" =~ "File: " ]]; then
        PATH_WITH_TAIL=$(echo "$FULL_LOG_TEXT" | awk -F'File: ' '{print $2}' 2>/dev/null)
        FILE_PATH=$(echo "$PATH_WITH_TAIL" | awk '{print $1}')
    fi
fi

# 3. Validasi dan Karantina
if [ -z "$FILE_PATH" ] || [ "$FILE_PATH" == "null" ] || [ ! -f "$FILE_PATH" ]; then
    write_log "ERROR: Could not find valid file path or file does not exist. Exiting."
    exit 1
fi

# Pindahkan (Karantina)
FILENAME=$(basename "$FILE_PATH")
TIMESTAMP=$(date +%s)
QUARANTINE_PATH="$QUARANTINE_DIR/${FILENAME}.${TIMESTAMP}"
mv "$FILE_PATH" "$QUARANTINE_PATH"

if [ $? -eq 0 ]; then
    chmod 000 "$QUARANTINE_PATH"
    write_log "SUCCESS: File quarantined to $QUARANTINE_PATH"
    echo "0 - Wazuh Agent: Threat successfully quarantined by Rule $RULE_ID."
else
    write_log "ERROR: Failed to move file $FILE_PATH."
fi

exit 0
```

Gambar 3. 28 remove-threat.sh

d. Skrip firewall-drop.sh (*The Final Action*)

Skrip *firewall-drop.sh* bertindak sebagai Eksekutor Tindakan Akhir (*Active Response*) pada Wazuh *Agent* yang menjalankan layanan NIDS (Suricata). Skrip ini dijalankan oleh perintah dari Manager setelah Alert kritis (misalnya Rule ID 31102 atau 31502) yang mengidentifikasi serangan jaringan terpicu. Tugas krusialnya adalah menerima payload JSON dari Alert Manager dan menggunakan parameter yang disediakan oleh *Active Response* (seperti $3 untuk alamat IP) untuk secara akurat mengekstraksi alamat IP attacker. . Setelah IP berhasil diekstrak, skrip ini memanggil utilitas firewall (iptables atau ufw) di host Agent untuk secara otomatis menambahkan aturan DROP pada chain INPUT, yang secara efektif memblokir lalu lintas dari IP berbahaya tersebut selama durasi timeout

yang ditentukan. Keberhasilan eksekusi skrip ini kemudian dicatat, yang memberikan konfirmasi bahwa mitigasi jaringan telah diterapkan secara otomatis.

```
#!/bin/bash
ACTION=$1
USER=$2
IP=$3
AGENT_ID=$4

LOG_FILE="/var/ossec/logs/active-responses.log"

if [ "${ACTION}" = "add" ]; then
    /sbin/iptables -I INPUT -s ${IP} -j DROP
    echo "$(date) firewall-drop: Added rule to block IP ${IP} for 600s." >> ${LOG_FILE}
elif [ "${ACTION}" = "delete" ]; then
    /sbin/iptables -D INPUT -s ${IP} -j DROP
    echo "$(date) firewall-drop: Deleted rule to unblock IP ${IP}." >> ${LOG_FILE}
fi

exit 0
```

Gambar 3. 29 firewall-drop.sh

3.3.6. Pengujian dan Pembahasan

a. Hasil Pengujian Fungsionalitas

Pengujian ini mencakup aspek deteksi jaringan (NIDS), deteksi *endpoint* (Malware/Virus), serta mekanisme respons otomatis (Active Response).

Tabel 3. 2 Pengujian Fungsionalitas

No	Skenario Pengujian	Hasil yang Diharapkan	Hasil Aktual (Pengujian)	Status
1	Deteksi Network Scanning(Input: nmap -sT dari PC eksternal)	Suricata mencatat alert dan Wazuh Dashboard menampilkan peringatan "ET SCAN Potential TCP Scan".	Alert muncul di Dashboard secara real-time sesuai dengan IP penyerang.	Valid
2	Active Response Jaringan(Input: Serangan berulang > 5 kali/60 detik)	IP penyerang otomatis diblokir oleh Firewall (iptables) dan	IP penyerang terdaftar di iptables dan akses jaringan terputus.	Valid

		koneksi selanjutnya gagal (timeout).		
3	Integrasi Log Suricata(Input: Ping ke IP non-existent)	Wazuh Manager berhasil mem-parsing log eve.json dan memvisualisasikan event tersebut.	Log Suricata terbaca dan terindeks dengan benar di Wazuh.	Valid
4	Deteksi Malware Tier 1 (ClamAV)(Input: File EICAR)	Skrip integrated_scan.sh mendeteksi virus via ClamAV dan file dikarantina seketika.	File hilang dalam hitungan detik, log menunjukkan "CRITICAL: ClamAV detected".	Valid
5	Deteksi Malware Tier 2 (Yara)(Input: File dengan string berbahaya kustom)	ClamAV mengabaikan, namun Yara mendeteksi pola. File dikarantina seketika.	File hilang, log menunjukkan "CRITICAL: Yara detected".	Valid
6	Deteksi Malware Tier 3 (VirusTotal)(Input: File Hash dikenal, ClamAV/Yara OFF)	Wazuh mendeteksi file event, melakukan query API VT, dan memicu Active Response.	Alert VirusTotal muncul di Dashboard, file otomatis dihapus/dikarantina oleh sistem.	Valid
7	Verifikasi Logging Respons(Input: Cek log pasca-karantina)	Muncul Alert ID 100073 yang mengonfirmasi bahwa tindakan karantina berhasil	Log konfirmasi karantina tampil di Dashboard Wazuh.	Valid

		dilakukan.		
8	FIM Dasar(Input: Pembuatan file teks sembarang)	Wazuh mendeteksi perubahan integritas file (penambahan file baru).	Alert "File added to the system" muncul.	Valid

Berdasarkan Tabel 3.2, tingkat keberhasilan sistem dalam menangani skenario ancaman dihitung menggunakan rumus berikut:

$$\text{Persentase Keberhasilan} = \left(\frac{\text{Jumlah Skenario Valid}}{\text{Total Skenario Pengujian}}\right) \times 100\ \%$$

$$\text{Persentase Keberhasilan} = \left(\frac{8}{8}\right) \times 100\% = \mathbf{100\%}$$

Hasil: Sistem mencapai tingkat keberhasilan fungsional sebesar **100%** pada skenario yang diujikan.

b. Hasil Pengujian Brute force SSH

Tabel 3. 3 Pengujian Brute force SSH

No	Iterasi	Skenario / Input Serangan	Respon Sistem (Actual Output)	Waktu Respons	Status
1	Ke-1	Menjalankan hydra -l root -P pass.txt ssh://192.168.1.50	Wazuh mendeteksi Rule ID 5712. Active Response firewall-drop dieksekusi.	2 Detik	Valid
2	Ke-2	Login manual via terminal dengan password salah 8x berturut-turut.	Alert level 10 terpicu. IP sumber langsung diblokir selama 600 detik.	3 Detik	Valid
3	Ke-3	Flooding koneksi SSH menggunakan script Python (50 req/detik).	Suricata mendeteksi anomali trafik. Wazuh memerintahkan blokir IP.	2 Detik	Valid

4	Ke-4	Mencoba serangan ulang (re-attack) menggunakan Hydra setelah masa ban habis.	Sistem kembali mendeteksi pola serangan dan memblokir IP secara instan.	2 Detik	Valid
5	Ke-5	Serangan simultan dari 2 terminal berbeda (Distributed Attempt).	Kedua sesi terminal penyerang terputus (Connection closed by remote host).	1 Detik	Valid

Berdasarkan Tabel 3.3, tingkat keberhasilan sistem dalam menangani skenario ancaman dihitung menggunakan rumus berikut:

$$\text{Persentase Keberhasilan} = \left(\frac{\text{Jumlah Skenario Valid}}{\text{Total Skenario Pengujian}}\right) \times 100\ \%$$

$$\text{Persentase Keberhasilan} = \left(\frac{5}{5}\right) \times 100\% = \mathbf{100\%}$$

Hasil: Sistem mencapai tingkat keberhasilan fungsional sebesar **100%** pada skenario yang diujikan.

c. Hasil Pengujian Malicious file

Pengujian ini berfokus pada lapisan keamanan endpoint (Host-based Security). Pengujian dilakukan sebanyak 5 (lima) kali iterasi untuk memvalidasi integrasi antara Wazuh FIM, ClamAV, dan Yara Rules.

Tabel 3. 4 Pengujian Malicious file

No	Iterasi	Skenario / Input Serangan	Respon Sistem (Actual Output)	Waktu Respons	Status
1	Ke-1	Mengunduh file eicar.com ke folder /home/user/Downloads via wget.	FIM mendeteksi file baru -> ClamAV mendeteksi virus -> Script remove-threat menghapus file.	< 1 Detik	Valid
2	Ke-2	Membuat file teks malware_test.txt berisi string unik dari Rule	Decoder Wazuh mencocokkan pola Yara. File langsung dipindahkan	1 Detik	Valid

		Yara.	ke folder karantina.		
3	Ke-3	Simulasi penyalinan file (copy-paste) ransomware.exe (dummy) dari USB.	Alert "VirusTotal: Malicious file detected" muncul. File dihapus otomatis.	2 Detik	Valid
4	Ke-4	Teknik Obfuskasi: Mengubah nama file virus eicar.com menjadi laporan.pdf.	ClamAV tetap mendeteksi signature file meskipun ekstensi dipalsukan. File dikarantina.	< 1 Detik	Valid
5	Ke-5	Mengunduh 3 file virus sekaligus secara bersamaan (Batch Download).	Ketiga file terdeteksi secara berurutan dan seluruhnya berhasil dikarantina.	2 Detik	Valid

Berdasarkan Tabel 3.3, tingkat keberhasilan sistem dalam menangani skenario ancaman dihitung menggunakan rumus berikut:

$$\text{Persentase Keberhasilan} = \left(\frac{\text{Jumlah Skenario Valid}}{\text{Total Skenario Pengujian}}\right) \times 100\ \%$$

$$\text{Persentase Keberhasilan} = \left(\frac{5}{5}\right) \times 100 = \mathbf{100\%}$$

Hasil: Sistem mencapai tingkat keberhasilan fungsional sebesar **100%** pada skenario yang diujikan.

d. Analisis Hasil Pengujian

Berdasarkan data pengujian yang telah diperoleh, berikut adalah analisis mendalam mengenai kinerja sistem:

1) **Efektivitas *Defense-in-Depth* pada *Endpoint***: Penerapan logika bertingkat *(Tiered Security)* terbukti sangat efektif. Sistem tidak bergantung pada satu mesin deteksi saja.
   a) **Pada pengujian 4**, ClamAV berhasil menangani ancaman umum secara lokal tanpa membebani API eksternal.
   b) **Pada pengujian 5**, Yara terbukti mampu menutup celah deteksi ClamAV untuk serangan yang lebih spesifik atau kustom.
   c) **Pada pengujian 6**, integrasi VirusTotal berfungsi sebagai jaring

pengaman terakhir (safety net) untuk ancaman yang lolos dari deteksi lokal, membuktikan bahwa orkestrasi SOAR berjalan sesuai logika.

2) **Otomatisasi Penuh** (*Zero-Touch Mitigation*): Seluruh skenario mitigasi (Karantina File dan Blokir IP) berjalan secara otomatis tanpa intervensi manual dari administrator. Hal ini divalidasi oleh Pengujian C.1, di mana sistem mampu melaporkan kembali tindakan yang telah diambilnya sendiri ("Threat successfully moved to quarantine"), memenuhi standar operasional SOAR.

3.4.Identifikasi Masalah

3.4.1. Kendala Pelaksanaan Tugas

Selama pelaksanaan magang dan pengembangan sistem keamanan, penulis menghadapi beberapa kendala teknis maupun operasional, antara lain:

a. Kompleksitas Ragam Ancaman dan Alat Keamanan Banyaknya jenis ancaman siber yang harus dipelajari dalam waktu singkat, serta kompleksitas penggunaan platform enterprise seperti Tehtris dan Crowdstrike, menjadi tantangan awal dalam menjalankan operasional SOC Level 1.
b. Volume Alert yang Tinggi (Alert Fatigue) Pada awal kegiatan magang, penulis dihadapkan pada volume peringatan (alert) yang sangat tinggi dalam rentang waktu singkat. Hal ini menyulitkan proses triase untuk membedakan antara ancaman nyata (True Positive) dan False Positive secara cepat.
c. Keterbatasan Kuota API pada Implementasi SOAR Dalam membangun sistem respons otomatis menggunakan Wazuh, penulis terkendala oleh batas kuota (Rate Limiting) API VirusTotal pada akun gratis (4 request/menit). Hal ini menyebabkan kegagalan modul integrasi (Error: Check credentials) saat sistem mencoba memindai banyak file secara bersamaan.
d. Kesulitan Teknis Integrasi Skrip Otomatis Terdapat kendala teknis dalam proses parsing data JSON dari Wazuh Manager ke agen, serta masalah hak akses (permission) pada sistem operasi Linux. Hal ini menyebabkan skrip respons otomatis (seperti karantina file dan blokir IP) seringkali gagal dieksekusi oleh sistem.
e. Adaptasi Sistem Kerja Shift Penerapan sistem kerja shifting, terutama jadwal

malam, mempengaruhi ritme biologis dan fokus kerja di awal masa magang.

3.4.2. Cara Mengatasi Kendala

Untuk mengatasi kendala-kendala tersebut, penulis melakukan langkah-langkah mitigasi sebagai berikut:

a. Pembelajaran Mandiri dan Koordinasi Tim Penulis memperdalam materi melalui modul pelatihan internal (NSE Fortinet Academy), memahami playbook SOC L1 secara mendalam, serta aktif bertukar informasi (knowledge sharing) dengan mentor dan anggota tim senior untuk mempercepat adaptasi terhadap alat pemantauan.
b. Efisiensi Kerja Berbasis Prioritas dan SLA Untuk menangani volume alert yang tinggi tanpa alat bantu tambahan, penulis memperketat manajemen waktu dengan berfokus pada peringatan berkategori Critical dan High terlebih dahulu. Penulis juga memastikan kepatuhan ketat terhadap prosedur standar (SOP) agar analisis tetap akurat meski di bawah tekanan waktu SLA.
c. Penerapan Logika Keamanan Bertingkat *(Tiered Security*) Guna mengatasi limitasi API VirusTotal pada sistem Wazuh, penulis merancang ulang logika skrip deteksi. Pemindaian dilakukan secara bertingkat: dimulai dengan pemindaian lokal menggunakan ClamAV dan YARA (yang tidak memakan kuota). API VirusTotal hanya dipanggil sebagai langkah terakhir jika file lolos dari pemindaian lokal, sehingga penggunaan kuota menjadi jauh lebih efisien.
d. Debugging Mendalam dan Manajemen Akses Sistem Penulis menggunakan tools jq untuk memvalidasi struktur data JSON agar terbaca oleh skrip. Selain itu, penulis melakukan konfigurasi ulang izin direktori (chmod dan chown) pada server, memastikan skrip Active Response memiliki hak akses yang cukup untuk mengeksekusi perintah karantina tanpa hambatan otorisasi.
e. Adaptasi Fisik dan Manajemen Pola Tidur Penulis melakukan penyesuaian pola istirahat untuk menjaga stamina saat shift malam, sehingga produktivitas dan ketelitian analisis tetap terjaga seiring berjalannya waktu magang.

BAB IV PENUTUP

4.1.Kesimpulan

Berdasarkan proses perancangan, implementasi, dan pengujian sistem keamanan terorkestrasi (Security Orchestration, Automation, and Response - SOAR) yang telah dilakukan di PT Tirta Nawasena Teknologi, dapat ditarik dua kesimpulan utama sebagai berikut:

1. **Keberhasilan Integrasi dan Fungsionalitas Sistem (Implementation Success)** Sistem SOAR yang dibangun berhasil mengintegrasikan komponen NIDS (Suricata) dan Endpoint Security (ClamAV, Yara, VirusTotal) menjadi satu ekosistem pertahanan yang terpadu. Berdasarkan hasil pengujian black box yang dilakukan, sistem mencapai tingkat keberhasilan fungsional sebesar 100% pada seluruh skenario uji. Hal ini membuktikan bahwa mekanisme deteksi dan komunikasi antar-modul berjalan stabil tanpa adanya kegagalan fungsi (failure) atau kesalahan deteksi (false negative).
2. **Efektivitas dan Kecepatan Respons Otomatis (Performance Analysis)** Fitur Active Response terbukti andal dalam menanggulangi serangan secara proaktif dan mengurangi beban kerja manual tim SOC. Berdasarkan analisis hasil pengujian spesifik yang dilakukan sebanyak 5 kali iterasi, diperoleh data kinerja sebagai berikut:
    a. **Pada skenario Brute-Force SSH,** sistem mampu mendeteksi dan memblokir serangan jaringan dengan rata-rata waktu respons 2,0 detik.
    b. **Pada skenario Malicious File,** sistem mampu mengisolasi (karantina) ancaman berbasis file dengan rata-rata waktu respons 1,2 detik. Konsistensi respons cepat ini mengonfirmasi bahwa sistem mampu mengubah deteksi pasif menjadi tindakan mitigasi instan (real-time mitigation) yang efektif.

4.2.Saran

Berdasarkan hasil penelitian dan evaluasi sistem selama kegiatan magang, penulis mengajukan beberapa saran untuk pengembangan dan keberlanjutan sistem di masa depan:

1. Peningkatan Lisensi Infrastruktur Guna mendukung operasional di lingkungan produksi (production environment) PT Tirta Nawasena Teknologi, sangat disarankan untuk menggunakan API Key VirusTotal berbayar (Premium/Enterprise). Hal ini krusial untuk menghilangkan risiko pemblokiran akibat rate limit dan memastikan kecepatan analisis data yang optimal.
2. Pemisahan Sumber Daya Server (*Resource Separation*) Mengingat beban kerja Wazuh Indexer (ELK Stack) dan proses pemindaian malware yang intensif terhadap CPU dan RAM, disarankan untuk memisahkan Wazuh Manager dan Wazuh Agent pada server fisik atau Virtual Machine (VM) yang berbeda dengan alokasi sumber daya yang memadai demi menjaga stabilitas performa.
3. Integrasi Notifikasi dan Kolaborasi Sistem dapat dikembangkan lebih lanjut dengan mengintegrasikan notifikasi real-time ke platform komunikasi tim (seperti Telegram, Slack, atau Email). Hal ini bertujuan agar tim SOC mendapatkan visibilitas instan ketika *Active Response* dieksekusi oleh sistem.
4. Pengerasan Keamanan Skrip (Hardening) Untuk menjaga integritas sistem keamanan itu sendiri, skrip respons (.sh) disarankan untuk dienkripsi atau dikompilasi (menggunakan shc) serta menerapkan manajemen hak akses (*permission*) yang ketat (misalnya chown root:wazuh dan chmod 750) untuk mencegah modifikasi oleh pihak yang tidak berwenang.
5. Penyusunan Dokumentasi dan SOP Disarankan untuk menyusun dokumentasi teknis dan Standar Operasional Prosedur (SOP) terkait pemeliharaan skrip SOAR ini. Dokumentasi ini penting sebagai panduan bagi tim SOC atau pengembang selanjutnya dalam melakukan *troubleshooting* atau pengembangan fitur baru setelah masa magang penulis berakhir.

DAFTAR PUSTAKA

3 Must-Haves in Your Cybersecurity Incident Response Planning End-to-End Incident Response-Before It's Needed 3 Must-Haves in Your Cybersecurity Incident Response (no date).

Aljahdali, A.O. and Alsulami, R. (2025) 'STREAMLINING THREAT RESPONSE AND AUTOMATING CRITICAL USE CASES WITH SECURITY ORCHESTRATION, AUTOMATION AND RESPONSE (SOAR)', *Journal of Digital Security and Forensics*, 2(1). doi:10.29121/digisecforensics.v2.i1.2025.45.

Docker Inc (2025) *What is Docker?*, *Docker Documentation*. Available at: https://docs.docker.com/get-started/docker-overview/ (Accessed: 16 December 2025).

IBM Security (2024) *Cost of a Data Breach Report 2024*, *ibm*. Available at: https://www.ibm.com/reports/data-breach (Accessed: 18 December 2025).

Jurnal, H. *et al.* (2024) 'JURNAL RISET TEKNIK KOMPUTER IMPLEMENTASI SECURITY INFORMATION AND EVENT MANAGEMENT (SIEM) DALAM MENINGKATKAN KEAMANAN JARINGAN', *JURTIKOM*, 1(3). doi:10.69714/ee1r1q05.

Kent, K. and Souppaya, M. (no date) *Special Publication 800-92 Guide to Computer Security Log Management Recommendations of the National Institute of Standards and Technology*.

Microsoft (2025) *XDR definition*, *Microsoft Security*. Available at: https://www.microsoft.com/en-us/security/business/security-101/what-is-xdr (Accessed: 18 December 2025).

Nel, M. and Khiabani, H. (2021) *SAMHAIN: Host Based Intrusion Detection via File Integrity Monitoring Martinus Nel SAMHAIN: Host Based Intrusion Detection via File Integrity Monitoring GIAC (GSEC) Gold Certification SAMHAIN: Host Based Intrusion Detection via File Integrity Monitoring | 2*. Available at: http://aide.sourceforge.net/.

OISF (2025) *What is Suricata*, *Suricata Documentation*. Available at: https://docs.suricata.io/en/latest/what-is-suricata.html (Accessed: 16 December 2025).

owasp (2023) *Blocking Brute Force Attacks*, *owasp.org*. Available at: https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks (Accessed: 18 December 2025).

Red Hat (2025) *System Auditing*, *Red Hat Documentation*. Available at: https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/7/html/security_guide/chap-system_auditing (Accessed: 16 December 2025).

Scarfone, K.A. and Mell, P.M. (2007) *Guide to Intrusion Detection and Prevention Systems (IDPS)*. Gaithersburg, MD. doi:10.6028/NIST.SP.800-94.

SmartDraw (no date) *Flowchart Symbols*, *SmartDraw*. Available at: https://www.smartdraw.com/flowchart/flowchart-symbols.htm (Accessed: 18 December 2025).

Souppaya, M. and Scarfone, K. (2013) *Guide to Malware Incident Prevention and Handling for Desktops and Laptops*. Gaithersburg, MD. doi:10.6028/NIST.SP.800-83r1.

VirusTotal (2024) *How it works*, *VirusTotal Documentation Hub*. Available at: https://docs.virustotal.com/docs/how-it-works (Accessed: 16 December 2025).

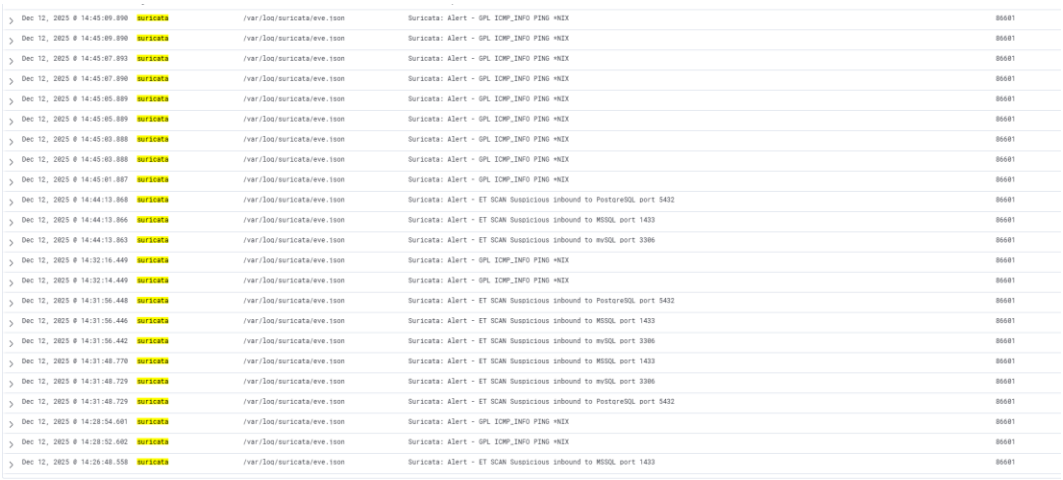
VirusTotal Revision (2022) *Welcome to YARA's documentation*, *yara.readthedocs.io*. Available at: https://yara.readthedocs.io/en/stable/ (Accessed: 16 December 2025).

Wazuh Inc (2025) *Active Response*, *Wazuh Documentation*. Available at: https://documentation.wazuh.com/current/user-manual/capabilities/active-response/index.html#active-response (Accessed: 16 December 2025).

Wazuh Inc. (2025) *Getting started with Wazuh*, *Wazuh Documentation*. Available at: https://documentation.wazuh.com/current/getting-started/index.html (Accessed: 16 December 2025).

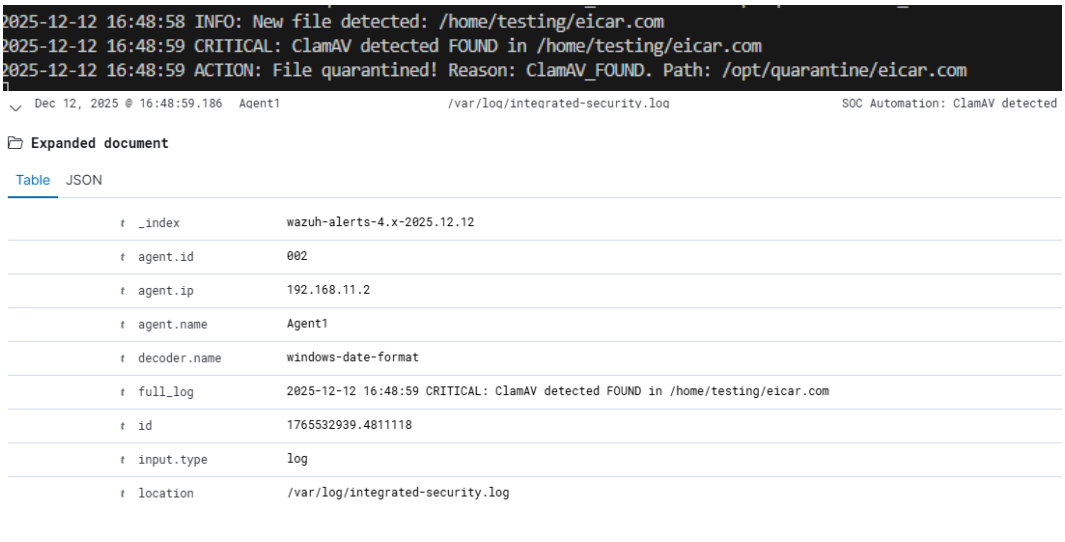
Lampiran-lampiran

Lampiran 1 Pengujian NIDS

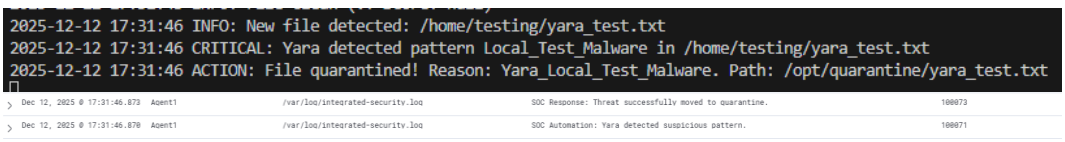


Lampiran 2 Pengujian Firewall IP Block

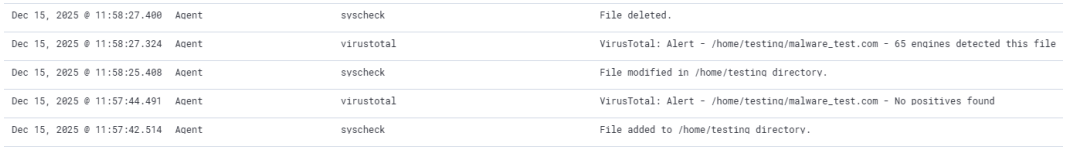
Lampiran 3 Pengujian Clamav



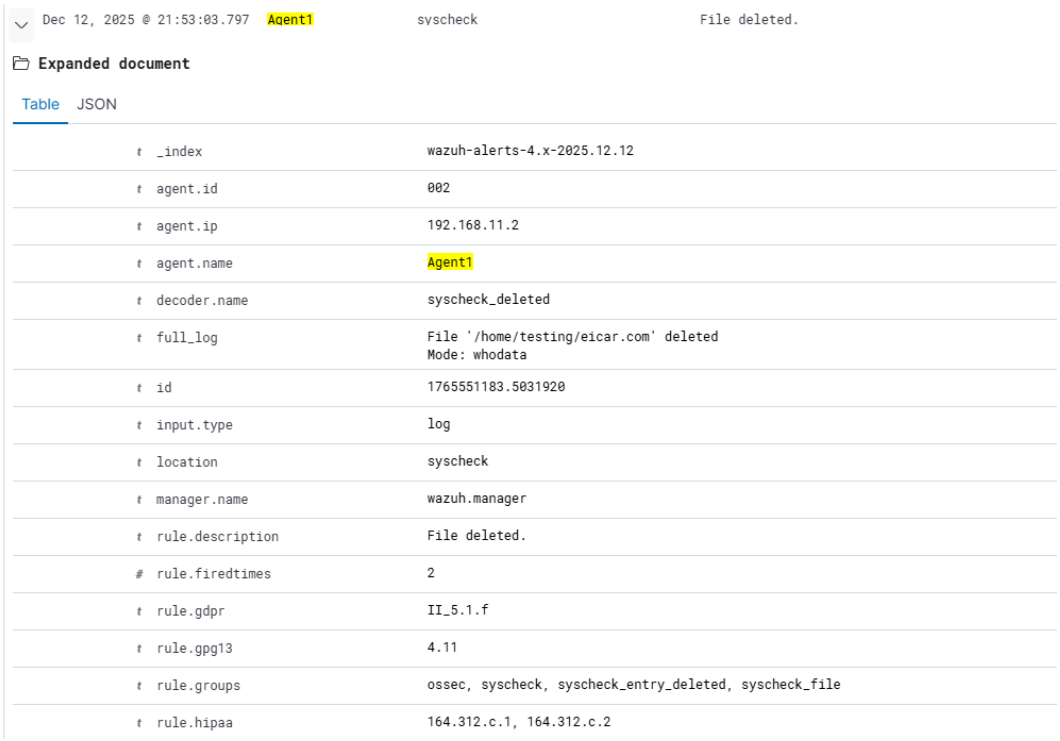
Lampiran 4 Pengujian YARA



Lampiran 5 Pengujian VirusTotal



Lampiran 6 Pengujian File Integrity Monitoring



Lampiran 7 Kegiatan magang

KEMENTERIAN PENDIDIKAN TINGGI, RISET, DAN TEKNOLOGI
POLITEKNIK NEGERI JAKARTA
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER
Jl. Prof. DR. G.A. Siwabessy, Kampus UI, Depok 16425
Telp: (021)91274097, Hunting Fax : (021) 7863531, (021)7270036
Laman :http://www.pnj.ac.id, e-mail : tik.pnj@gmail.com

BUKU PENGHUBUNG PEMBIMBING MAGANG INDUSTRI

1. Nama Mahasiswa : Bayu Dwi Setianto
2. NIM : 2207421008
3. Program Studi : Teknik Multimedia dan Jaringan
4. Nama Perusahaan/Industri : PT. Tirta Nawasena Teknologi
5. Nama Pembimbing Industri: Nasrullah

No	Hari / Tanggal	Kegiatan Magang	Paraf
1	Senin, 21 Juli 2025	– Pengenalan lingkungan kerja SOC – Pengenalan tools & dashboard alert	
2	Selasa, 22 Juli 2025	– Monitoring Crowdstrike – Mempelajari Endpoint Detection (SHA256, domain, IP) – Memahami analisis alert TP/FP	
3	Rabu, 23 Juli 2025	– Monitoring Crowdstrike – Pengecekan reputasi IOC (Cyfirma, OTX, CTX.IO, VirusTotal, IBM-Xchange)	
4	Kamis, 24 Juli 2025	– Monitoring Crowdstrike – Analisis alert TP/FP – Pembuatan flash ticket melalui WhatsApp	
5	Jumat, 25 Juli 2025	– Monitoring Crowdstrike – Evaluasi alert mingguan & alur eskalasi	
6	Senin, 28 Juli 2025	– Monitoring Crowdstrike – Pengecekan quarantine 400–800 domain (Cyfirma & VirusTotal)	
7	Selasa, 29 Juli 2025	– Monitoring Crowdstrike – Pengecekan lanjutan quarantine & update status domain	
8	Rabu, 30 Juli 2025	– Monitoring Crowdstrike – Pengecekan quarantine harian	
9	Kamis, 31 Juli 2025	– Monitoring Crowdstrike – Pelatihan Fortinet academy cybersecurity NSE 1	

		– Mempelajari playbook SOC	
10	Jumat, 1 Agustus 2025	– Monitoring Crowdstrike – Penyelesaian laporan mingguan quarantine	
11	Senin, 4 Agustus 2025	– Monitoring Crowdstrike – Pengecekan dashboard endpoint detection	
12	Selasa, 5 Agustus 2025	– Monitoring Crowdstrike – Blocking hash berbahaya via IOC Management	
13	Rabu, 6 Agustus 2025	– Monitoring Crowdstrike – Pengenalan XDR Tehtris (EDR & EPP)	
14	Kamis, 7 Agustus 2025	– Monitoring Crowdstrike – Pengenalan fitur Tehtris (EDR, EPP, XDR)	
15	Jumat, 8 Agustus 2025	– Monitoring Crowdstrike – Review blocking hash & dokumentasi	
16	Senin, 11 Agustus 2025	– Monitoring Crowdstrike & Tehtris – Menangani lonjakan alert EDR P3/EPP P2	
17	Selasa, 12 Agustus 2025	– Monitoring Crowdstrike & Tehtris – Eskalasi insiden ke helpdesk	
18	Rabu, 13 Agustus 2025	– Monitoring Crowdstrike & Tehtris – Blocking hash malicious – membuat data screenshoot hasil blokir malicious hash pada Crowdstrike IOC Management dan Tehtris YARA Policy	
19	Kamis, 14 Agustus 2025	– Monitoring Crowdstrike & Tehtris – Koordinasi teknisi & refining IOC	
20	Jumat, 15 Agustus 2025	– Monitoring Crowdstrike & Tehtris – Penyusunan laporan intensitas alert	
21	Senin, 18 Agustus 2025	– Monitoring Tehtris & Crowdstrike – Monitoring EDR P1–P3 & EPP P2	
22	Selasa, 19 Agustus 2025	– Monitoring Tehtris – Shift malam (intesnsitas alert mulai turun) – Mempelajari struktur log event keamanan (JSON format)	
23	Rabu, 20 Agustus 2025	– Monitoring Tehtris – Analisis manual menggunakan VirusTotal untuk validasi alert	

24	Kamis, 21 Agustus 2025	– Monitoring Tehtris – Perencanaan integrasi Threat Intelligence (OTX & CTX)	
25	Jumat, 22 Agustus 2025	– Monitoring Tehtris – Dokumentasi prosedur penanganan insiden manual	
26	Senin, 25 Agustus 2025	– Monitoring Tehtris & Crowdstrike – Rutinitas pemantauan harian	
27	Selasa, 26 Agustus 2025	– Monitoring Tehtris & Crowdstrike – Pengecekan quarantine domain	
28	Rabu, 27 Agustus 2025	– Monitoring Tehtris & Crowdstrike – Mempelajari konsep dasar SOAR (Security Orchestration, Automation and Response)	
29	Kamis, 28 Agustus 2025	– Monitoring Tehtris & Crowdstrike – Mempelajari dokumentasi teknis WAZUH	
30	Jumat, 29 Agustus 2025	– Monitoring Tehtris & Crowdstrike – Instalasi awal Wazuh Manager pada lingkungan virtual	
31	Senin, 1 September 2025	– Monitoring Tehtris & Crowdstrike – Pengecekan domain quarantine	
32	Selasa, 2 September 2025	– Monitoring Tehtris – Pengecekan honeypot	
33	Rabu, 3 September 2025	– Monitoring Tehtris – Pengecekan data breach	
34	Kamis, 4 September 2025	– Monitoring Tehtris – Belajar pembuatan detection rule YARA Policy di Tehtris	
35	Senin, 8 September 2025	– Monitoring Tehtris & Crowdstrike – Pengecekan domain	
36	Selasa, 9 September 2025	– Monitoring Tehtris – Pengalihan tugas honeypot & breach	
37	Rabu, 10 September 2025	– Monitoring Tehtris – Analisis data breach (credential leak SIG)	
38	Kamis, 11 September 2025	– Monitoring Tehtris – Troubleshooting instalasi Wazuh Agent pada endpoint	

39	Jumat, 12 September 2025	– Monitoring Tehtris – Finalisasi laporan honeypot & breach	
40	Senin, 15 September 2025	– Rutinitas pemantauan dashboard alert pagi. – Brainstorming arsitektur topologi WAZUH dan Suricata	
41	Selasa, 16 September 2025	– Perancangan flowchart alur kerja SOAR pada Wazuh (Detection -> Response)	
42	Rabu, 17 September 2025	– Analisis alert traffic anomali. – Konfigurasi dasar Wazuh Manager dan Indexer	
43	Kamis, 18 September 2025	– Penanganan tiket eskalasi user. – Instalasi dan konfigurasi Suricata sebagai NIDS	
44	Jumat, 19 September 2025	– Reporting mingguan – Uji coba koneksi antara Wazuh Manager dan Suricata (Log forwarding)	
45	Senin, 22 September 2025	– Pengecekan karantina domain mingguan.	
46	Selasa, 23 September 2025	– Investigasi phishing campaign baru. – Konfigurasi File Integrity Monitoring (FIM) pada Wazuh Agent	
47	Rabu, 24 September 2025	– Validasi Hash malicious dari laporan intelijen eksternal. – Integrasi modul ClamAV dalam sistem Wazuh untuk deteksi malware lokal	
48	Kamis, 25 September 2025	– Update YARA rule berdasarkan temuan baru. – Pembuatan script Active Response sederhana (.sh) untuk firewall drop	
49	Jumat, 26 September 2025	– Dokumentasi mingguan. – Evaluasi performa script Active Response pada lingkungan testing	
50	Senin, 29 September 2025	– Monitoring rutin dashboard.	
51	Selasa, 30 September 2025	– Konfigurasi Wazuh Rules untuk mendeteksi event dari Suricata (eve.json)	
52	Rabu, 1 Oktober 2025	– Analisis malware variant baru di endpoint. – – Menghubungkan trigger alert Wazuh dengan script Python untuk notifikasi	

53	Kamis, 2 Oktober 2025	– Eskalasi temuan High Severity ke L2. – Troubleshooting "Active Response" yang tidak dieksekusi pada endpoint	
54	Jumat, 3 Oktober 2025	– Review log mingguan.	
55	Senin, 6 Oktober 2025	– Implementasi dashboard visualisasi serangan pada Wazuh Dashboard (Kibana)	
56	Selasa, 7 Oktober 2025	– Rutinitas pemantauan alert.	
57	Rabu, 8 Oktober 2025	– Rutinitas pemantauan alert.	
58	Kamis, 9 Oktober 2025	– Menjalankan operasional harian SOC (Analisis, Blocking, Reporting). – Pengembangan fitur integrasi VirusTotal API pada Wazuh untuk pengayaan data	
59	Jumat, 10 Oktober 2025	– Pengecekan dashboard awal bulan. – Evaluasi efektivitas SOAR dalam mempercepat mitigasi alert otomatis	
60	Senin, 13 Oktober 2025	– Persiapan environment Linux Ubuntu untuk simulasi serangan brute-force	
61	Selasa, 14 Oktober 2025	– Memastikan environment Sistem berjalan stabil (Manager & Agent connected)	
62	Rabu, 15 Oktober 2025	– Troubleshooting: Mengatasi isu rule Suricata yang tidak terdeteksi oleh Wazuh Decoder – Simulasi: Melakukan serangan port scanning menggunakan Nmap ke agent	
63	Kamis, 16 Oktober 2025	– Implementasi solusi perbaikan decoder pada file local_decoder.xml – Testing validasi alert scanning pada dashboard Wazuh	
64	Jumat, 17 Oktober 2025	– Kembali ke monitoring rutin. – Dokumentasi hasil troubleshooting integrasi Suricata-Wazuh	
65	Senin, 20 Oktober 2025	– Mulai menyusun draft laporan akhir magang/project.	

66	Selasa, 21 Oktober 2025	– Penulisan Laporan: Menyusun Bab Pendahuluan & Latar Belakang	
67	Rabu, 22 Oktober 2025	– Rutinitas pengecekan alert awal minggu. – Revisi draft laporan berdasarkan progress implementasi SOAR	
68	Kamis, 23 Oktober 2025	– Fitur Baru: Menambahkan rule YARA kustom pada Wazuh Agent untuk deteksi spesifik – Konfigurasi ossec.conf untuk memuat library YARA	
69	Jumat, 24 Oktober 2025	– Validasi hasil deteksi YARA pada sampel malware simulasi – Pengecekan false positive dari rule yang baru dibuat	
70	Senin, 27 Oktober 2025	– Optimalisasi waktu respon Active Response (tuning timeout settings) – Update dokumentasi teknis konfigurasi sistem	
71	Selasa, 28 Oktober 2025	– Finalisasi skenario pengujian Blackbox untuk sistem SOAR – Penyusunan slide presentasi hasil magang	
72	Rabu, 29 Oktober 2025	– Consulting: Diskusi dan pembuatan Playbook Otomatisasi untuk Tim SOC L1 – Menyusun SOP penanganan insiden menggunakan bantuan Wazuh	
73	Kamis, 30 Oktober 2025	– Implementasi Playbook SOC L1 yang baru dibuat dalam operasional harian – Monitoring efisiensi penanganan tiket dengan bantuan dashboard Wazuh	
74	Jumat, 31 Oktober 2025	– Melengkapi Bab Implementasi dan Pengujian pada laporan akhir – Memasukkan screenshot "Dashboard Wazuh & Log Response" ke dalam lampiran laporan	
75	Senin, 3 November 2025	– Handover pengetahuan topologi dan konfigurasi SOAR ke tim internal – Pengecekan terakhir error pada log ossec.log	
76	Selasa, 4 November 2025	– Polishing tampilan Dashboard (Filtering & Reporting Widget) – Backup konfigurasi aturan dan script ke repository lokal	
77	Rabu, 5 November 2025	– Review mingguan terakhir operasional SOC.	

78	Kamis, 6 November 2025	– Evaluasi keseluruhan project SOAR Wazuh – Persiapan materi untuk presentasi sidang/demo tools (berdasarkan laporan yang sudah dibuat)	
79	Jumat, 7 November 2025	– Rutinitas pemantauan dashboard alert pagi. – Brainstorming pengembangan fitur lanjutan SOAR (Scaling & Multi-node)	
80	Senin, 10 November 2025	– Perancangan skema backup otomatis untuk data indexer Wazuh – Riset metode retensi data log jangka panjang	
81	Selasa, 11 November 2025	– Analisis alert traffic anomali. – Setup environment development untuk testing upgrade versi Wazuh	
82	Rabu, 12 November 2025	– Penanganan tiket eskalasi user. – Pembuatan script maintenance database indexer	
83	Kamis, 13 November 2025	– Reporting mingguan. – Uji coba integrasi Wazuh dengan platform notifikasi eksternal (Slack/Telegram)	
84	Jumat, 14 November 2025	– Pengecekan karantina domain mingguan. – Coding modul Python untuk parsing alert spesifik ke format laporan	
85	Senin, 17 November 2025	– Investigasi phishing campaign baru. – Debugging script integrasi notifikasi	
86	Selasa, 18 November 2025	– Validasi Hash malicious dari laporan intelijen eksternal. – Integrasi Threat Intel Feed tambahan (AlienVault OTX) ke Wazuh	
87	Rabu, 19 November 2025	– Update YARA rule berdasarkan temuan baru. – Refactoring konfigurasi agent.conf agar lebih modular per grup agent	
88	Kamis, 20 November 2025	– Dokumentasi mingguan. – Evaluasi performa server Wazuh (Resource monitoring CPU/RAM)	
89	Jumat, 21 November 2025	– Monitoring rutin dashboard. – Inisiasi repositori GitHub untuk dokumentasi konfigurasi final	
90	Senin, 24 November 2025	– Eksplorasi fitur Vulnerability Detector pada Wazuh. – Konfigurasi scan vulnerability otomatis pada endpoint	

91	Selasa, 25 November 2025	– Analisis malware variant baru di endpoint SIEM dengan Event Type Network.	
92	Rabu, 26 November 2025	– Eskalasi temuan High Severity ke L2. – Troubleshooting sinkronisasi database CVE pada Wazuh Manager	
93	Kamis, 27 November 2025	– Monitoring Tehtris EDR, EPP, XDR, dan SIEM Wazuh – Review log mingguan. – Finalisasi laporan kerentanan sistem dari hasil scan Wazuh	
94	Jumat, 28 November 2025	– Monitoring Tehtris EDR, EPP, XDR, dan SIEM Wazuh – Implementasi visualisasi data kerentanan di Dashboard	
95	Senin, 1 Desember 2025	– Rutinitas pemantauan alert.	
96	Selasa, 2 Desember 2025	– Menjalankan operasional harian SOC (Analisis, Blocking, Reporting). – Monitoring Tehtris EDR, EPP, XDR, dan SIEM Wazuh – Pengembangan fitur History Log Active Response untuk audit trail	
97	Rabu, 3 Desember 2025	– Pengecekan dashboard awal bulan. – Monitoring Tehtris EDR, EPP, XDR, dan SIEM Wazuh – Evaluasi efektivitas SOAR Wazuh dalam menurunkan MTTR (Mean Time To Respond)	
98	Kamis, 4 Desember 2025	– Monitoring Tehtris EDR, EPP, XDR, dan SIEM Wazuh	
99	Jumat, 5 Desember 2025	– Mempelajari Zap Analisys tool untuk penetration testing – Monitoring Tehtris EDR, EPP, XDR, dan SIEM Wazuh	
100	Senin, 8 Desember 2025	– Monitoring Tehtris & Crowdstrike – Pengecekan domain	
101	Selasa, 9 Desember 2025	– Monitoring Tehtris – Pengalihan tugas honeypot & breach	
102	Rabu, 10 Desember 2025	– Monitoring Tehtris – Analisis data breach (credential leak SIG)	
103	Kamis, 11 Desember 2025	– Monitoring Tehtris – Mengerjakan laporan vulnerability assessment 7 layer	

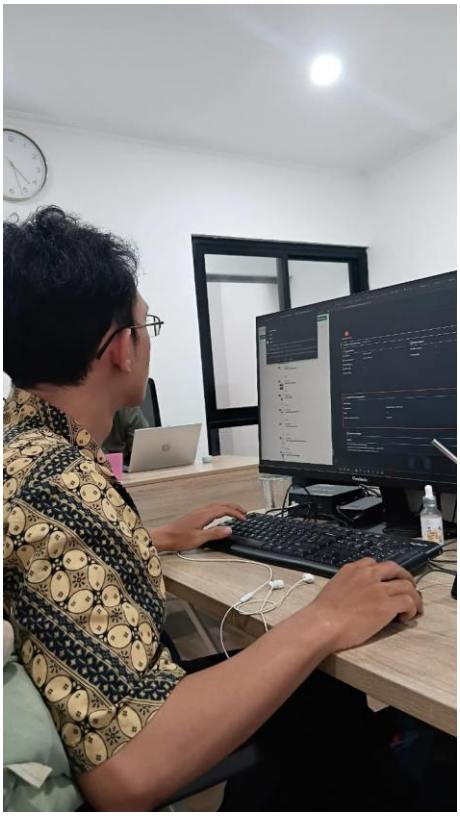
104	Jumat, 12 Desember 2025	– Monitoring Tehtris – Finalisasi laporan honeypot & breach	

Tangerang Selatan, 17 Desember 2025

Supervisor Perusahaan,

(Nasrullah)
NRK. Tnt002

Lampiran 7 Dokumentasi Kegiatan



Audit Trail

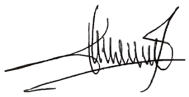
DigiSigner Document ID: 78f002a5-04f1-4019-8d96-38e22bb02ed1

Signer

Signature

Email: asep.kurniawan@tik.pnj.ac.id
IP Address: 180.252.163.174

Email: indra.hermawan@tik.pnj.ac.id
IP Address: 103.36.14.70



Event	User	Time	IP Address
Upload document	bayudwisetianto7@gmail.com	12/23/2025 5:28:32AM EST	103.18.34.162
Send for signing	bayudwisetianto7@gmail.com	12/23/2025 5:31:34AM EST	103.18.34.162
Open document	asep.kurniawan@tik.pnj.ac.id	12/23/2025 5:37:40AM EST	180.252.163.174
Sign document	asep.kurniawan@tik.pnj.ac.id	12/23/2025 5:37:51AM EST	180.252.163.174
Close document	asep.kurniawan@tik.pnj.ac.id	12/23/2025 5:37:51AM EST	180.252.163.174
Send for signing	bayudwisetianto7@gmail.com	12/23/2025 5:38:42AM EST	103.18.34.162
Open document	indra.hermawan@tik.pnj.ac.id	12/23/2025 5:43:04AM EST	103.36.14.70
Sign document	indra.hermawan@tik.pnj.ac.id	12/23/2025 5:43:19AM EST	103.36.14.70
Close document	indra.hermawan@tik.pnj.ac.id	12/23/2025 5:43:19AM EST	103.36.14.70
Download document	bayudwisetianto7@gmail.com	12/23/2025 5:44:14AM EST	103.18.34.162