

Diseño e Implementación de una plataforma XSOAR en Docker: Automatización de detección y respuesta a incidentes en el laboratorio de Ciberseguridad

Cuadrado Avilés, Santiago Alejandro

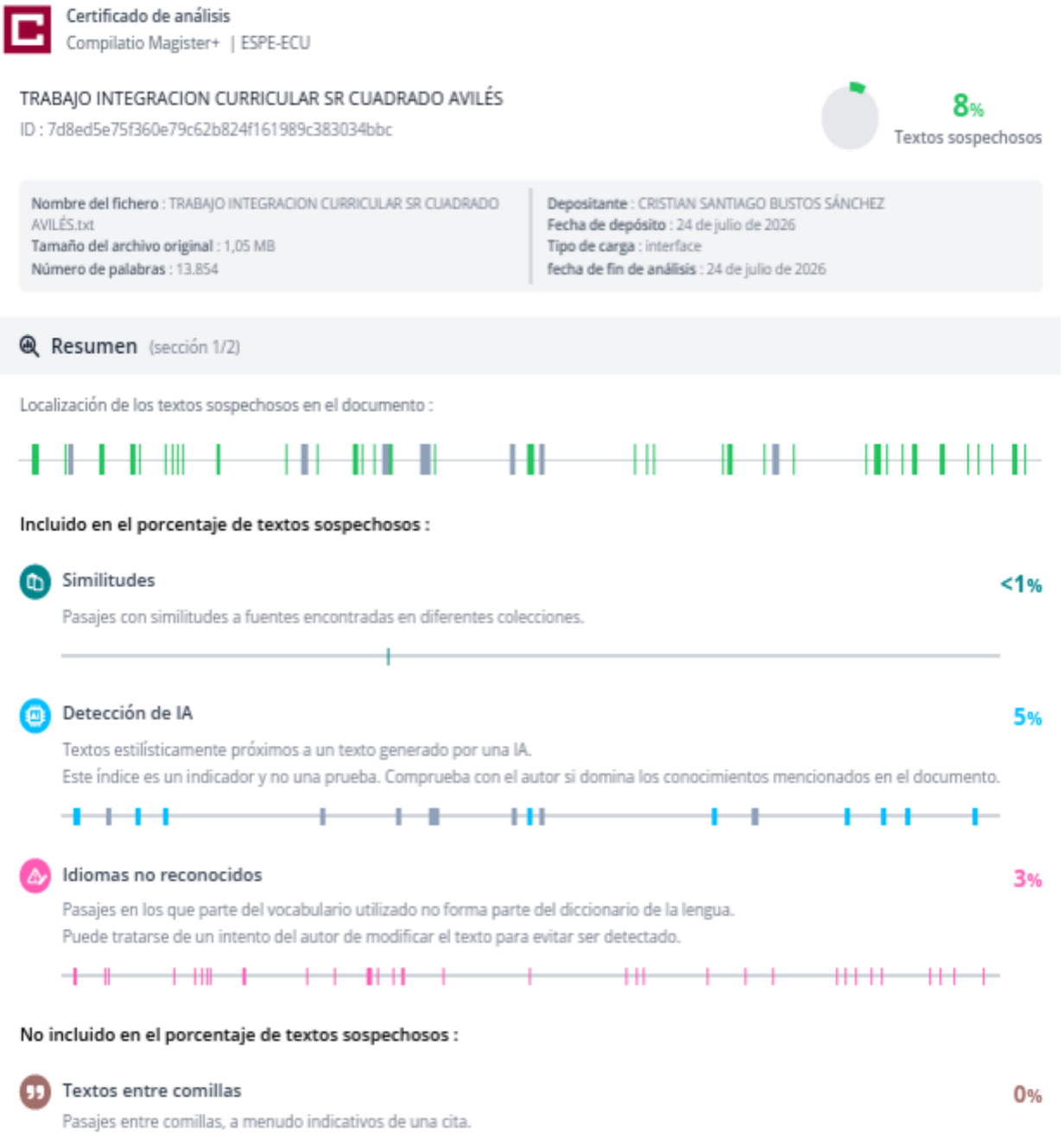
Departamento de Ciencias de la Computación

Carrera de Tecnologías de la Información

Trabajo de integración curricular, previo a la obtención del título de Ingeniero en Tecnologías de la Información

Ing. Bustos Sánchez Cristian Santiago, PhD.

24 de julio de 2026



Validar únicamente en FirmaEC.
Firmado electrónicamente por:
CRISTIAN SANTIAGO BUSTOS SANCHEZ

Ing. Bustos Sánchez Cristian Santiago, PhD

Director de Trabajo de Titulación

Departamento de Ciencias de la Computación

Carrera de Tecnologías de la Información

Certificación

Certifico que el trabajo de integración curricular: "**Diseño e Implementación de una plataforma XSOAR en Docker: Automatización de detección y respuesta a incidentes en el laboratorio de Ciberseguridad**" fue realizado por el señor Cuadrado Avilés, Santiago Alejandro**,** el mismo que cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE; habiéndose efectuado una revisión integral de forma y de fondo conforme a los lineamientos y formatos institucionales vigentes. Además, fue revisado y analizado en su totalidad por la herramienta de prevención y/o verificación de similitud de contenidos; razón por la cual me permito acreditar y autorizar para que se lo sustente públicamente.

Sangolquí, 24 de julio de 2026

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
CRISTIAN SANTIAGO BUSTOS SANCHEZ

Ing. Bustos Sánchez Cristian Santiago, PhD

Director de Trabajo de Titulación

C.C.: 1500863111

Departamento de Ciencias de la Computación

Carrera de Tecnologías de la Información

Responsabilidad de Autoría

Yo, Cuadrado Avilés, Santiago Alejandro, con cédula de ciudadanía n° 1724177835, declaro que el contenido, ideas y criterios del trabajo de integración curricular**: Diseño e Implementación de una plataforma XSOAR en Docker: Automatización de detección y respuesta a incidentes en el laboratorio de Ciberseguridad** es de mi autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 24 de julio de 2026

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
SANTIAGO ALEJANDRO CUADRADO AVILES

Cuadrado Avilés Santiago Alejandro

C.C.: 1724177835

Departamento de Ciencias de la Computación

Carrera de Tecnologías de la Información

Autorización de Publicación

Yo Cuadrado Avilés, Santiago Alejandro, con cédula de ciudadanía n°1724177835, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de integración curricular: **Título: Diseño e Implementación de una plataforma XSOAR en Docker: Automatización de detección y respuesta a incidentes en el laboratorio de Ciberseguridad** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 24 de julio de 2026

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
SANTIAGO ALEJANDRO CUADRADO AVILES

Cuadrado Avilés Santiago Alejandro

C.C.: 1724177835

Dedicatoria

Quiero comenzar esta dedicatoria hacia Dios, ya que sin el nada de esto podría ser posible, y por guiarme siempre, aunque todo parecía estar en contra.

A mis padres, Vinicio y Gabriela, que con sus enseñanzas y esfuerzo hicieron demasiado por mí. Un hombre que nunca bajó los brazos y, a pesar de las adversidades, siempre se mantiene de pie como un guerrero que siempre admiraré y respetaré. Por otro lado, la mujer que me dio la vida es digna de admirar; sin importar cómo se sintiera, siempre hace lo posible por verme feliz y estar conmigo en todo momento.

Para mi hermana Meliza, que ahora está cumpliendo sus sueños al otro lado del mundo, pero siempre está pendiente de mí. Cabe mencionar que la extraño y agradezco todas las veces que vio a su hermano menor y que, a pesar de la distancia, esto de igual forma es para ella.

A mi pareja, Dayana, y a mi hijo, Mathias: ustedes hacen que yo me levante todos los días y, desde el momento en que estamos juntos, hacen mis días más bonitos con sus ocurrencias y risas. Debo mencionarles que siempre los voy a amar, que voy a ver por su bienestar y espero nunca hacerles faltar nada.

A mis abuelos Hugo, Anita y, en especial, a Enrique y Rebeca, que, así sea con un consejo o algún gusto que me supieron dar, siempre lo recordaré, ya que de igual manera hicieron posible este momento. A mis tíos Jorge y Magaly, que siempre me hicieron sentir como un hijo más dentro de sus vidas y me apoyaron con mucho más que palabras durante este tiempo universitario.

A mi hermano Daniel, que, aunque ya no se encuentra conmigo de forma física, sé que desde el cielo está feliz por verme cumplir mis sueños. Quiero decirle que lo extraño con mi alma, que siempre será mi confidente, mi complemento y, nunca lo voy a dejar de amar.

Un abrazo hasta el cielo, hermano. Te amo.

Agradecimiento

Agradezco a la Universidad de las Fuerzas Armadas ESPE por brindarme la oportunidad de formarme profesionalmente y por todas las experiencias adquiridas durante esta etapa. De igual manera, expreso mi gratitud a los docentes que compartieron sus conocimientos y contribuyeron a mi crecimiento académico y personal.

A mi tutor, por su paciencia, orientación y tiempo dedicado durante el desarrollo de este trabajo. Sus observaciones, recomendaciones y apoyo constante fueron fundamentales para superar las dificultades presentadas y culminar esta etapa de manera satisfactoria.

A la familia de mi pareja, por abrirme las puertas de su hogar y acogerme como a un hijo más. Gracias por brindarme siempre una mano, hacerme sentir parte de su familia y apoyarme con cariño en los momentos en que más lo necesité. Cada gesto y palabra de aliento fueron significativas para continuar hasta alcanzar esta meta.

A mis amigos, especialmente a Ariel y Angelo, por los buenos momentos, las experiencias compartidas y aquellos recuerdos que permanecerán siempre en mi memoria. También agradezco a mi primo Juanito, quien con sus ocurrencias y alegría hizo que los días fueran más llevaderos durante esta etapa.

Finalmente, agradezco a todas las personas que estuvieron presentes y que, de una u otra manera, me impulsaron a continuar hasta cumplir este objetivo

Índice de contenidos

Certificación 3
Responsabilidad de Autoría 4
Autorización de Publicación 5
Dedicatoria....6
Agradecimiento....7
Resumen 13
Abstract....14
Capítulo I 15
Introducción 15
Planteamiento del problema....16
Formulación del problema o pregunta de investigación....16
Objetivos....17
Objetivo general 17
Objetivos específicos....17
Justificación 18
Capítulo II 19
Estado del arte....19
Marco teórico 23
Marco conceptual....30
Capítulo III 34
Metodología 34
Enfoque de investigación....34
Alcance de la investigación 34

Método de investigación y desarrollo....34
Diseño de investigación....35
Población y muestra....35
Técnicas e instrumentos....37
Validación, confiabilidad y consideraciones éticas....37
Diseño físico de la arquitectura....38
Diseño Lógico de la Arquitectura....39
Construcción del entorno base....41
Despliegue de servicios contenerizados....44
Integración entre TheHive, Cortex y MISP....48
Integración entre Wazuh y TheHive....51
Configuración de respuesta activa y cuarentena....53
Configuración de escenarios automatizados....54
Capítulo IV....58
Análisis de los resultados....58
Resultados Generales de los escenarios evaluados....58
Resultados del escenario de carga de archivos sospechosos.....59
Resultados del tiempo medio de detección....60
Resultados del tiempo medio de respuesta....62
Resultados del nivel de automatización.....63
Resultados de clasificación.....65
Resultados del consumo de CPU....67
Resultados del consumo de memoria....68
Resultados del escenario de fuerza bruta SSH....70
Resultados del escenario de movimiento no autorizado de un archivo protegido.....71
Resultados de las pruebas de persistencia y recuperación....73

Discusión de los resultados....74
Comparación del tiempo medio de detección entre los distintos niveles de carga....75
Comparación del tiempo medio de respuesta entre los distintos niveles de carga....75
Niveles de Automatización alcanzado....76
Clasificación de los resultados....76
Comparación del consumo promedio de CPU según el nivel de carga....76
Comparación del consumo promedio de memoria según el nivel de carga....77
Cumplimiento de las verificaciones de persistencia y recuperación....77
Capítulo V....78
Conclusiones....78
Recomendaciones....79
Trabajos Futuros....80
Referencias:....81

Índice de tablas

Tabla 1 Componentes incluidos en la muestra técnica y su rol 36

Tabla 2 Escenarios automatizados definidos para la validación del entorno 54

Tabla 3 Resultados generales de los escenarios evaluados. 58

Tabla 4 Distribución de archivos sospechosos en cada nivel 59

Tabla 5 Tiempo medio de detección según los niveles de carga. 60

Tabla 6 Resultados del tiempo de respuesta según el nivel de carga. 62

Tabla 7 Actividades consideradas para evaluar el nivel de automatización. 64

Tabla 8 Resultados de las pruebas de efectividad de la detección. 65

Tabla 9 Resultados del consumo de CPU según el nivel de carga. 67

Tabla 10 Resultados del consumo de memoria según el nivel de carga. 68

Tabla 11 Resultados del escenario de fuerza bruta SSH 70

Tabla 12 Resultados del escenario de movimiento no autorizado de un archivo protegido. 71

Tabla 13 Resultados de las pruebas de persistencia y recuperación 73

Índice de figuras

Figura 1 Infraestructura física del sistema 39

Figura 2 Diseño lógico del sistema 40

Figura 3 Dashboard de Wazuh levantado en el sistema. 41

Figura 4 Vulnerabilidades analizadas en Wazuh Dashboard 43

Figura 5 Agente Ubuntu monitoreado en Wazuh. 43

Figura 6 Contenedores del entorno encendidos. 45

Figura 7 Interfaces web de TheHive 46

Figura 8 Interfaces web de Cortex 46

Figura 9 Interfaces web de MISP 47

Figura 10 Conexión de Cortex configurada en TheHive. 49

Figura 11 Analizador MISP habilitado en Cortex. 50

Figura 12 Configuración de integración entre Wazuh y TheHive. 52

Figura 13 Evidencia de cuarentena generada por la respuesta activa. 54

Figura 14 Comparación del tiempo de detección de los distintos niveles de carga 61

Figura 15 Comparación del tiempo de respuesta de los distintos niveles de carga 63

Figura 16 Porcentajes de actividades ejecutadas de forma automática. 65

Figura 17 Distribución de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos 66

Figura 18 Comparación del consumo de CPU según el nivel de carga 68

Figura 19 Comparación del consumo de memoria según el nivel de carga 69

Figura 20 Cumplimiento de las verificaciones realizadas después del reinicio. 74

Resumen

La gestión de eventos de ciberseguridad puede presentar demoras y dificultades en el momento en que las herramientas de detección, análisis y respuesta funcionan de forma independiente y necesitan actividades manuales. Frente a esta problemática, la presente investigación tiene como propósito diseñar e implementar un entorno de orquestación y automatización para la respuesta ante incidentes en el laboratorio de Ciberseguridad, como referencia en las capacidades funcionales de XSOAR junto con las herramientas Wazuh, TheHive, Cortex y MISP. La investigación adoptó un enfoque tecnológico aplicado, con alcance exploratorio, descriptivo y cuasiexperimental. La arquitectura fue aplicada sobre una máquina virtual Ubuntu Server y utilizó Docker y Docker Compose para el despliegue de los servicios en contenedores. Se trabajó con escenarios controlados de archivos sospechosos, fuerza bruta SSH y movimiento no autorizado de un archivo protegido. Durante las cinco ejecuciones del escenario de archivo sospechoso el tiempo medio de detección obtenido es de 0,007 segundos, 15,734 segundos de tiempo medio de respuesta y 16,407 segundos de tiempo promedio de extremo a extremo. Para la prueba de fuerza bruta SSH el tiempo de respuesta registrado fue de 15,844 segundos. A su vez, se obtuvieron cinco verdaderos positivos en las pruebas de efectividad, por último, las actividades consideradas en los flujos evaluados fueron ejecutadas de manera automática. Durante las pruebas de carga, los 35 eventos fueron gestionados de manera correcta, sin que se evidencien interrupciones por tiempo de espera, Así mismo, las pruebas de persistencia corroboran con la recuperación de los nueve servicios contenerizados y recuperación del flujo luego de los reinicios. Los resultados indican que el entorno planteado funciona de manera correcta para realizar prácticas controladas. No obstante, los resultados obtenidos se encuentran limitados a los escenarios y necesidades evaluadas.

***Palabras clave**:* SOAR, MTTD, MTTR, XSOAR

Abstract

When detection, analysis, and response tools begin to function independently and necessitate manual tasks, managing cybersecurity events can cause delays and challenges. Using the functional capabilities of XSOAR and the tools Wazuh, TheHive, Cortex, and MISP as references, the current study attempts to address this problem by designing and implementing an orchestration and automation environment for incident response in the cybersecurity laboratory. A technologically applied strategy with an experimental and descriptive scope was created for the study. The architecture was implemented on a virtual machine running Ubuntu Server, and services were deployed in containers using Docker and Docker Compose. Unauthorized movement of a secured file, SSH brute force, and controlled scenarios of suspicious files were investigated. The suspicious file scenario was executed five times, with an average detection time of 0.007 seconds, an average reaction time of 15.734 seconds, and an average end-to-end time of 16.407 seconds. The recorded response time for the SSH brute force test was 15.844 seconds. At the same time, the efficacy tests yielded five true positives, and ultimately, the actions taken into account in the assessed flows were carried out automatically. The 35 events were handled properly during the load tests, and there was no indication that waiting time caused any disruptions. Similarly, the flow recovery following restarts and the recovery of the nine containerized services are verified by the persistence tests. The findings show that the suggested setting is operating as intended for carrying out regulated procedures. Nevertheless, the outcomes are restricted to the demands and circumstances that were assessed.

***Keywords**:* SOAR, MTTD, MTTR, XSOAR

Capítulo I

Introducción

La ciberseguridad se ha transformado con el paso de los años convirtiéndose en un punto fundamental para para la protección de los sistemas, servicios, redes y de la información relevante. El incremento de los ataques ha generado la necesidad de utilizar mecanismos efectivos para la gestión de incidentes y su pronta detección. Dentro de esta perspectiva, surgen retrasos en la identificación y atención de situaciones críticas por el uso de alertas tradicionales, apoyo en procesos manuales y mecanismos aislados.

Frente a esta problemática, las herramientas SOAR, conocidas como Security Orchestration, Automation and Response, indican una opción tecnológica orientada a la integración de herramientas de seguridad, la optimización de tareas y la gestión de flujos de respuesta.

El Trabajo de Integración Curricular se enfoca en el diseño, ejecución y validación de un entorno de orquestación, automatización y respuesta a incidentes de ciberseguridad con la utilización de las capacidades funcionales de XSOAR y puesta en ejecución a través de contenedores Docker en el laboratorio de Ciberseguridad. El proyecto contempla la incorporación de plataformas especializadas como Wazuh, Cortex, TheHive y MISP.

El manejo de Docker en esta propuesta conforma un entorno contenerizado e interoperable, esta tecnología favoreció el despliegue de componentes en un espacio controlado. Docker facilita el empaquetado de las aplicaciones junto con sus dependencias para una ejecución consistente en distintos entornos. Este atributo es esencial para la implementación de una arquitectura experimental, ya que favorece el aislamiento de los servicios, la gestión de volúmenes persistentes, la definición de redes y la organización estructural de las herramientas seleccionadas.

Planteamiento del problema

En los entornos de laboratorio de ciberseguridad, la gestión de eventos de seguridad exige procedimientos eficientes que favorecen la detección, el análisis y la respuesta ante posibles amenazas. No obstante, mientras estos procesos dependen de acciones manuales y de herramientas que no se encuentran integradas, la atención de los eventos se vuelve lenta, fragmentada y menos eficaz.

Esta problemática dificulta la correlación de la información procedente de distintas fuentes, debido a que cada herramienta funciona de manera diferente y requiere la intervención del usuario en la revisión de alertas, la validación de datos, la generación de casos y la aplicación de acciones de respuesta. Esta situación genera retrasos en la identificación de eventos, duplica las tareas y aumenta los errores durante la gestión de incidentes de seguridad.

La carencia de un entorno centralizado de orquestación y automatización dentro del laboratorio de ciberseguridad, limita la capacidad para reproducir procesos operativos similares a los que están establecidos en el centro de operaciones de seguridad. Esta situación dificulta la coordinación de herramientas disponibles y limita la validación de la automatización de los procesos de detección, análisis y respuesta en escenarios simulados.

Por lo que, dentro de un entorno controlado de laboratorio, la problemática se enfoca en la ausencia de una plataforma orientada a la automatización y coordinación de la gestión de eventos de seguridad, con el propósito de mejorar la eficiencia operativa y reforzar la capacidad de respuesta a las amenazas de ciberseguridad.

Formulación del problema o pregunta de investigación

¿Cómo influye la falta de integración y automatización de las herramientas de ciberseguridad en la gestión de eventos de seguridad en el laboratorio, y de qué forma la implementación de un entorno de orquestación mejorar el análisis, articulación y respuesta frente a amenazas?

Objetivos

Objetivo general

Diseñar, implementar y validar un entorno de orquestación y automatización basado en funcionalidades similares a las de XSOAR y apoyado en contenedores Docker, con la incorporación de herramientas de ciberseguridad para llevar a cabo procesos de detección, análisis y respuesta ante incidentes de seguridad en el laboratorio de ciberseguridad.

Objetivos específicos

- Analizar las bases teóricas y técnicas asociadas con XSOAR, Docker y las herramientas de ciberseguridad elegidas, mediante una revisión documental para determinar los criterios esenciales para su incorporación en el laboratorio.
- Diseñar una arquitectura que permita organizar los componentes del entorno, tomando en cuenta la comunicación entre servicios, el manejo de volúmenes persistentes y los requisitos de despliegue para favorecer la integración y disponibilidad de las plataformas implementadas.
- Implementar la integración entre Wazuh, TheHive, Cortex y MISP a través de APIs y configuraciones, para automatizar los procesos de detección, análisis y respuesta ante eventos de seguridad.
- Evaluar la funcionalidad del entorno implementado mediante escenarios simulados y métricas relacionadas con la efectividad de detección, los tiempos de detección y de respuesta, el nivel de automatización, el comportamiento bajo carga, el consumo de recursos y la recuperación de los servicios tras un reinicio, para determinar su desempeño dentro del laboratorio de ciberseguridad.

Justificación

Este estudio se justifica por la necesidad de un ambiente para la administración de eventos de seguridad en el laboratorio, Al no contar con un entorno centralizado los procesos referentes a la atención de amenazas dependen de herramientas que operan de forma independiente y con procesos manuales, lo que reduce la articulación entre componentes y complica la implementación de respuestas.

Desde una perspectiva tecnológica, este plan aporta el diseño y desarrollo de un sistema ligero e independiente tras el uso de Docker, orientado a la conformación de un ambiente controlado, aislado y reproducible. Esto favorece la administración de la arquitectura y las validaciones sin comprometer los servicios existentes en el laboratorio.

Dentro del área académica, el estudio es oportuno debido a que ofrece un ambiente de experimentación para adquirir experiencia relacionada con la ciberseguridad, manejo de servicios y arquitectura de sistemas. Cabe indicar que, al ser una implementación en un espacio real de laboratorio, resulta útil como base para prácticas, análisis y mejoras posteriores dentro de la Universidad.

Capítulo II

Estado del arte

Durante los últimos años, la gestión de incidentes de ciberseguridad ha adquirido mayor relevancia debido al incremento de amenazas informáticas, la variedad de sistemas tecnológicos y la necesidad de minimizar el tiempo de respuesta ante eventos que puedan dañar la continuidad de los servicios. Bajo este contexto, la respuesta ante eventualidades no solo es una actividad luego de un ataque, sino más bien es un proceso relacionado con la preparación, detección, análisis, contención y recuperación. Por lo cual, NIST expone que la respuesta a incidentes se tiene que integrar con la gestión de las amenazas de ciberseguridad porque esta integración permite reducir el impacto de los eventos y que la capacidad de reacción de las organizaciones sea mejor (Nelson, Rekhi, Souppaya, & Scarfone, 2025).

En los centros de operaciones de seguridad, uno de los problemas más frecuentes identificados son la cantidad de alertas emitidas por diferentes fuentes de monitoreo. Dichas alertas pueden proceder de sistemas SIEM, agentes, soluciones EDR, plataformas de inteligencia de amenazas u otros servicios de seguridad. Si no existe una adecuada organización, la evaluación de estos eventos se convierte en un proceso lento, repetitivo y dependiente de revisiones manuales. Por lo cual, se ha propuesto la necesidad de que las tareas se automaticen, la información se consolide y se disminuya el cambio continuo entre herramientas en el transcurso de la investigación de incidentes (Pulyala, Desetty, & Jangampet, 2019).

Las plataformas SOAR aparecen como una respuesta a esta carencia. Norem, Rice, Oesch y Weber analizan las herramientas SOAR en entornos SOC e indican que dichas soluciones mejoran y agilizan la atención de alertas a través de funciones de incorporación, automatización y ayuda al operador. Los autores mencionados enfatizan que la adopción de SOAR no solo busca valorarse según el número de acciones que automatiza, sino también por

la utilidad real que genera para los analistas y por su capacidad para unirse con el flujo de trabajo del grupo de seguridad (Norem, y otros, 2021).

Para complementar, Bridges y otros autores analizan el caso práctico de herramientas SOAR y mencionan que las investigaciones SOC necesitan la consulta de varias fuentes, la vinculación de registros y el reporte de resultados en sistemas de tickets o gestión de sucesos. El estudio muestra que las plataformas SOAR incrementan la eficiencia y disminuyen el cambio de contexto durante la investigación, a pesar de que advierten que una excesiva automatización puede afectar la efectividad del análisis si no se llega a combinar con el criterio del analista. Este concepto es clave para la presente investigación porque la plataforma propuesta busca apoyar el análisis y la respuesta, sin la necesidad de eliminar la intervención humana (Bridges, y otros, 2022).

La unión entre herramientas representa otro rasgo significativo dentro de los entornos SOAR. Sworna, Islam y Babar exponen que la conexión de las herramientas de seguridad se realiza con APIs, plugins y scripts. No obstante, la identificación y utilización de las APIs esenciales representa una tarea compleja para los equipos SOC, puesto que existe una gran variedad de formatos, documentación y estructuras de cada solución. Este precedente se vincula con el presente trabajo, debido a que la comunicación entre Wazuh, TheHive, Cortex y MISP requiere de claves API, scripts de integración y parámetros de configuración ideales (Sworna, Islam, & Babar, 2022).

Con respecto a las herramientas de código abierto, Kalonji elabora una tesis basada en las capacidades de Cortex en el proyecto DYNAMO. Aquí se presenta a Cortex como un instrumento orientado al análisis de observables y la aplicación de acciones de respuesta por medio de analizadores y respondedores. También se analiza la conexión con TheHive y MISP, al demostrar que estas plataformas admiten una integración dentro del proceso de respuesta a incidentes. Esto es fundamental porque diferencia el rol que cumple cada herramienta (Kalonji, 2022).

La investigación de Kalonji establece que Cortex cumple una función distinta a la gestión de casos, porque su función principal se basa en el procesamiento de indicadores técnicos, como son las direcciones IP, dominios, URL, archivos o hashes. Es fundamental tener en cuenta esto ya que la arquitectura propuesta en este proyecto utiliza Cortex como un componente de análisis automático y no solo como un repositorio de incidentes.

Otro precedente para este proyecto es la investigación de Pereira, la cual sugiere una arquitectura de detección y respuesta frente a ciberataques a través de herramientas de código abierto como por ejemplo Wazuh, TheHive, Cortex, MISP y Shuffle. Su enfoque propuesto contempla la emisión de alertas desde Wazuh, la generación de casos en TheHive, la evaluación de artefactos por medio de Cortex y el enriquecimiento de información mediante MISP. Esta investigación se relaciona con el presente trabajo porque demuestra la viabilidad de la orquestación de herramientas de código abierto dentro de un flujo de detección, análisis y respuesta (Pereira V. R., 2023).

Pereira destaca que Cortex examina artefactos como direcciones IP, dominios y hashes de archivos, lo cual mejora el proceso investigativo. Además, TheHive centraliza la recepción y organización de las alertas, mientras que MISP aporta información de inteligencia de amenazas.

Según algunos trabajos de integración TheHive se considera una alternativa de código abierto para la gestión de casos de seguridad. La plataforma mencionada centraliza alertas, organiza observables, registra tareas y garantiza la trazabilidad durante la investigación. El registro oficial de TheHive muestra que dentro de un caso admite observables, tareas, TTPs y archivos adjuntos, elementos que estructuran la información vinculada con una actividad sospechosa. (Groenewegen, 2021).

En cuanto a MISP, la documentación oficial lo muestra como una plataforma de inteligencia de amenazas especializada en compartir, almacenar y analizar la información. MISP gestiona indicadores de compromiso, eventos, atributos y taxonomías, esto facilita la

estructuración de datos sobre incidentes o amenazas (MISP, 2022). Para este trabajo, su favorece el análisis de los observables junto con la información registrada previamente, lo que fortalece el proceso de ampliación de información adicional (Wagner, Dulaunoy, Wagener, & Iklody, 2016).

Wazuh es fundamental para esta investigación por sus capacidades SIEM y XDR. La documentación oficial indica que la plataforma posee componentes como agente, servidor, indexer y dashboard, así mismo, contempla casos vinculados con el análisis de logs, identificación de archivos sospechosos, seguimiento de integridad de archivos, detección de vulnerabilidades, respuesta a incidentes y cumplimiento normativo. Esto respalda su uso como fuente de eventos y alertas en la arquitectura orientada a la automatización de respuestas (Wazuh, 2018).

Docker y Docker Compose se relacionan con trabajos de investigación y proyectos de laboratorio en ciberseguridad, puesto que estas tecnologías despliegan diversos servicios en una sola infraestructura de forma ordenada (Irvine, Thompson, McCarrin, & Khosalim, 2017). Docker Compose define aplicaciones multicontenedor mediante archivos de configuración, lo que ayuda en la creación, ejecución y administración de servicios interdependientes. Para entornos académicos esta característica resulta apropiada, donde se necesita reproducir arquitecturas compuestas sin la necesidad de instalar cada componente directo en el sistema operativo (Chouliaras, Kantzavelou, Maglaras, Pantziou, & Ferrag, 2023).

A partir de investigaciones realizadas con antelación, se reconocen tres líneas esenciales vinculadas con el presente trabajo. En primer lugar, el uso de plataformas SOAR permite optimizar la atención de alertas en centro de operaciones de seguridad. En segundo lugar, corresponde a la integración de herramientas de código abierto como Wazuh, TheHive, Cortex y MISP para detección, análisis y enriquecimiento de eventos. Por último, se asocia con la implementación de entornos controlados a través del uso de contenedores.

A pesar de que existen proyectos y trabajos de investigación previos, estos se enfocan en componentes específicos o escenarios de pruebas particulares. Por este motivo, la presente investigación busca diferenciarse al aplicar una arquitectura integrada en un laboratorio académico, con el uso de herramientas open source y contenedores Docker para validar un flujo funcional de detección, análisis y respuesta ante eventos de ciberseguridad.

Marco teórico

Ciberseguridad y gestión de incidentes

La ciberseguridad comprende un conjunto de tecnologías y procesos dirigidos a proteger sistemas, redes e información ante sucesos no permitidos, interrupciones o daños. Su objetivo abarca la prevención de ataques y la gestión de incidentes de seguridad para la reducción de sus consecuencias. Bajo este contexto, la ciberseguridad se entiende como un procedimiento constante que incluye prevención, monitoreo y mejora continua (A. Nelson, 2025).

La gestión de incidentes es parte del proceso, porque organiza la atención de eventos que exponen la confidencialidad e integridad de los recursos tecnológicos. Identificar el evento, examinar el alcance, priorizar la atención, registrar los resultados alcanzados forman parte de una gestión adecuada. Esta organización reduce la dependencia de acciones improvisadas y favorece la aplicación de procesos estructurados (A. Nelson, 2025).

Para este trabajo, la gestión de incidentes mantiene relación con la capacidad de registrar las alertas emitidas por herramientas de monitoreo, ordenarlas dentro de una plataforma de casos, evaluarlas mediante servicios especializados y mejorarlas con información de inteligencia de amenazas. Por tal motivo, la arquitectura plantead trasciende la instalación individual de herramientas y articula componentes que fortalecen el tratamiento de eventos de seguridad.

Eventos, alertas e incidentes de seguridad

Los eventos de seguridad hacen referencia a un suceso observable en un sistema, red, aplicación o servicio. Este suceso puede ser normal, sospechoso o malicioso, según el contexto en el que ocurre. No todos los eventos son incidentes, por lo que, todo evento requiere un análisis previo a su clasificación y a la definición de una acción de respuesta. (A. Nelson, 2025).

Una alerta se considera una notificación producida por una herramienta de seguridad cuando tiene una condición definida, como la ejecución de una regla, la identificación de una vulnerabilidad, un comportamiento irregular o la identificación de un indicador sospechoso. Las alertas advierten al analista sobre eventos que necesitan revisión, aunque por sí solas no generan un contexto suficiente para tomar decisiones.

Un incidente de seguridad comprende una afección real o potencial sobre los recursos tecnológicos. Puede tener relación con un malware, accesos no autorizados, fuga de información o actividades que involucran la seguridad del entorno. La diferencia entre evento, alerta e incidente define un flujo ordenado para la gestión de seguridad: en primer lugar, se produce un evento, posterior se produce una alerta y, por último, si se presenta un riesgo relevante, se gestiona como un incidente (A. Nelson, 2025).

Plataformas SOAR

Estas plataformas combinan capacidades de orquestación, automatización y respuesta ante incidentes de seguridad. Al hablar de orquestación se menciona la conexión entre diferentes herramientas dentro de un flujo habitual; la automatización por otro ejecuta tareas recurrentes mediante reglas o playbooks; y la respuesta se enfoca en gestionar acciones que posibiliten contener, analizar o reducir el impacto de la anomalía (Norem, y otros, 2021).

Este tipo de plataformas son claves porque los equipos trabajan con grandes cantidades de alertas y distintas fuentes de información. En el omento que las herramientas

operan de manera independiente, el analista tiene que revisar varias consolas, copiar información de forma manual y registrar los resultados en sistemas separados. SOAR reduce esta división mediante el acoplamiento de servicios y la aplicación de flujos de trabajo establecidos (Bridges, y otros, 2022).

El valor de una plataforma SOAR no está solo en automatizar tareas, sino en consentir que la información circule entre herramientas. Como, por ejemplo, una alerta producida por un SIEM se puede convertir en un caso, complementarse con inteligencia de amenazas, a través de un motor externo y accionar una acción de respuesta. De tal forma, esta plataforma hace que la investigación sea más ordenada y minimiza la carga operativa (Sworna, Islam, & Babar, 2022) .

XSOAR y playbooks

XSOAR representa una plataforma basada en orquestación, automatización y respuesta de seguridad. Su funcionalidad se sustenta en integraciones, flujos de trabajo y playbooks que definen acciones ante determinados tipos de eventos. Dentro de una arquitectura de seguridad, XSOAR actúa como elemento de coordinación de tareas, conexión de servicios y ejecución de procesos previamente definidos (Guduru, 2022).

Los playbooks expresan series de acciones que orientan el tratamiento de una alerta o incidente. Estas acciones incluyen obtención de información, análisis de observables, consulta de inteligencia de amenazas, categorización del evento, asignación de tareas e implementación de medidas de respuesta. Su utilidad radica en la estandarización de procedimientos, la reducción del tiempo de atención y la conservación de la trazabilidad (Guduru, 2022).

En esta investigación, el estudio de XSOAR explica la estructura utilizada por las plataformas SOAR para la coordinación y ejecución de actividades. Aunque el producto comercial no hizo parte del despliegue práctico sus principios encaminaron la organización de flujos, condiciones de decisión y comunicación entre componentes

SIEM y Wazuh

El sistema SIEM facilita la recopilación, correlación y análisis de eventos de seguridad procedentes de distintas fuentes. Gestionar registros, aplicar normas de detección y generar alertas en el caso de que se reconozcan condiciones importantes, forma parte de la tarea principal. En una arquitectura de seguridad informática, el SIEM actúa como un centro de información capaz de identificar conductas sospechosas.

Wazuh es una plataforma de seguridad informática de código abierto que ofrece funcionalidades SIEM y XDR. Esto incluye agentes que se instalan en endpoints, un sistema que se encarga de la gestión de eventos, un indexador que acumula información y un dashboard que presenta los resultados, mientras que la plataforma gestiona registros, detección de malware, monitoreo de integridad de archivos, vulnerabilidades y respuesta a incidentes (Wazuh, 2018).

Para la presente investigación, Wazuh cumple como una fuente de detección y generación de alertas. Los agentes recolectan la información desde los sistemas monitoreados, a la vez que el servidor procesa la información por medio de reglas y decodificadores. Este rol es esencial porque la arquitectura SOAR requiere eventos procedentes de una fuente de monitoreo como punto inicial del análisis y la respuesta (Wazuh, 2018).

TheHive

Es una plataforma enfocada en la gestión de alertas, casos y observables asociados con incidentes de seguridad. Su propósito consiste en que los analistas ordenen información, designen tareas, documenten hallazgos y continúen el seguimiento durante la investigación. En contraste con un SIEM, TheHive no solo se centra en recolectar grandes dimensiones de logs, sino también en estructurar información para la atención de eventos (Amine, 2023).

TheHive reúne información relacionada con alertas, incidentes o actividades inusuales. Conforme con su documentación oficial, dichos casos admiten observables, tareas, TTPs y

archivos adjuntos. Esto posibilita que el analista resguarde evidencia, asocie indicadores y registre el avance de la investigación en la misma plataforma (Amine, 2023).

En función a la arquitectura propuesta, TheHive funciona como un eje de organización y seguimiento. Resulta clave porque recibe alertas, las transforma en casos, relaciona observables y conserva resultados de análisis. Por ende, desempeña un rol indispensable dentro del flujo de seguridad, puesto que transforma una alerta técnica en un elemento gestionable dentro del proceso de respuesta (Amine, 2023).

Cortex

Es una plataforma enfocada en el análisis automatizado de observables técnicos. Ejecuta analizadores y respondedores sobre direcciones IP, dominios, URL, hashes y archivos. Su objetivo es respaldar la investigación de sucesos a través de consultas a fuentes internas o externas que aporten información extra sobre un indicador (Kalonji, 2022).

Los analizadores de Cortex consultan servicios y bases de datos para obtener resultados técnicos sobre los observables. Este resultado puede incorporar coincidencias, reputación, contexto o clasificación. Los respondedores, por otro lado, ejecutan acciones determinadas sobre un observable o caso, de acuerdo con las condiciones configuradas. Esta competencia convierte a Cortex en una pieza útil para automatizar tareas de análisis dentro de un flujo de respuesta (Kalonji, 2022).

Para la investigación, Cortex no sustituye a TheHive ni a MISP. Ya que TheHive ordena alertas y casos; MISP se encarga de almacenar y vincular inteligencia de amenazas; y Cortex efectúa análisis sobre los observables. Esta diferencia es crucial para eludir confusiones entre funciones de cada herramienta y para entender el modelo de integración planteado.

MISP

Se considera una plataforma de inteligencia de amenazas de código abierto dirigida al almacenamiento, relación, análisis y el flujo de información. MISP gestiona indicadores de compromiso e información relacionada con amenazas. Su finalidad es simplificar que los

equipos de seguridad intercambien y consulten información organizada sobre amenazas (Wagner C. D., 2016).

MISP organiza la información en eventos, atributos, taxonomía, galaxias y objetos. Estos eventos recopilan información relacionada con amenazas e incidentes, los datos técnicos como direcciones IP, dominios, URL o hashes considerados como atributos, las taxonomías hacen que la clasificación de información sea más fácil y las galaxias (MISP, 2022).

MISP es la fuente de la inteligencia de amenazas en la arquitectura propuesta. Al integrar MISP, se garantiza que el observable no solo se analice de forma aislada, sino se compare con los indicadores registrados. Esta función proporciona contexto al análisis y sustenta la decisión sobre la relación del elemento técnico con amenazas conocidas (Borges, 2021).

Docker y Docker Compose

Docker es una tecnología basada en contenedores destinados a la ejecución aislada y portable de aplicaciones junto con sus dependencias. En contraste con lo que es una instalación tradicional, en donde cada servicio se configura en el sistema operativo, los contenedores posibilitan la separación de elementos y disminuyen problemas entre dependencias. Dicha característica permite la administración de entornos compuestos por diversos servicios (Merkel, 2018).

Docker Compose integra este enfoque al facilitar la definición de aplicaciones multicontenedor por medio de un archivo de configuración. La documentación oficial muestra que Compose controla y administra aplicaciones compuestas por distintos contenedores, lo que posibilita su ejecución y organización (Docker, 2018).

Integración mediante APIs

La integración de plataformas de seguridad necesita mecanismos que faciliten el envío, la recepción y el análisis de información entre servicios. En un entorno SOAR ocurre mediante APIs, conectores, webhooks, scripts o plugins. De esta manera, una alerta generada por una

herramienta se envía hacia otra plataforma para su análisis. Las APIs ejecutan un rol primordial ya que posibilitan que las herramientas no actúen de forma aislada. Mediante ellas, un servicio consulta información, envía datos, ejecuta procesos y recupera resultados. Para el proyecto, las APIs facilitan la conexión con Wazuh, TheHive, Cortex y MISP en el flujo dirigido a detección, análisis y respuesta (Sworna, Islam, & Babar, 2022).

El funcionamiento de la arquitectura requiere una configuración adecuada de las APIs, las credenciales de autenticación y los formatos de intercambio. Si los componentes no se definen de forma correcta, las herramientas pueden encontrarse implementadas y disponibles, aunque no intercambien información entre ellas.

Métricas de detección y respuesta

Estas métricas permiten examinar el comportamiento de una plataforma de seguridad ante escenarios definidos. En una arquitectura SOAR, las métricas permiten determinar si la automatización genera beneficios en cuanto al tiempo, menos tareas manuales y coherencia de los resultados. Entre las métricas que se usan de forma habitual se encuentra el tiempo medio de detección, el tiempo medio de respuesta y el porcentaje alcanzado (Srinithi, 2026).

El tiempo medio de detección analiza el lapso entre la emisión de un evento y su identificación proveniente de la herramienta de monitoreo. El tiempo medio de respuesta consiste en el intervalo entre la detección del evento y la implementación de una acción de análisis o respuesta. Dichas métricas posibilitan sustituir una descripción general del funcionamiento por una evaluación sustentada en evidencias.

Marco conceptual

Ciberseguridad

La ciberseguridad hace referencia a la protección de sistemas, redes, equipos y programas contra ataques maliciosos, esto a través de buenas prácticas, procesos detallados y tecnologías orientadas a evitar accesos no autorizados, pérdidas de información, interrupciones en el servicio y alteraciones en los equipos (Cisco, 2017).

Incidentes de Seguridad.

Los incidentes de seguridad se definen como cualquier situación o evento que ponga en riesgo la confidencialidad y la integridad de los datos y que vulnere la Política de Seguridad de la Información (Banco Santander, 2020).

SOAR

SOAR busca coordinar soluciones de seguridad y automatizar los diferentes procedimientos para brindar respuestas eficientes ante ciberataques. Esto no solo permite responder rápido frente sino también comprender y prevenir futuros incidentes (Palo Alto Networks, 2024).

XSOAR

XSOAR se entiende como una plataforma que lleva la orquestación, automatización extendida y respuestas de seguridad más allá de un SOAR tradicional, al incluir funciones avanzadas para gestionar casos y una automatización más profunda. Esto permite aumentar la eficiencia y el tiempo de respuesta (Kafami, 2025).

Playbooks

Los Playbooks o libros de estrategias son listas detalladas que contienen una secuencia de acciones definidas y se ejecutan ante condiciones determinadas. Su principal función es estandarizar los procesos y reducir las tareas repetitivas (TecnoSolucion, 2022).

API

Una API (Interfaz de Programación de Aplicaciones) es un grupo de reglas que permiten intercambiar datos e interactuar de manera fundamentada, por lo tanto, se lo considera como un intermediario que permite a dos programas comunicarse entre sí mediante un conjunto de protocolos. Las API son necesarias para crear aplicaciones de software modernas que necesitan conectar y compartir información (Oracle, 2025).

SIEM

La gestión de información y eventos (SIEM) es una tecnología que ayuda a las organizaciones a identificar y afrontar posibles amenazas y vulnerabilidades de seguridad antes de que estas interrumpan las operaciones, también se la utiliza para recopilar y analizar información de seguridad que viene de diferentes fuentes (IBM, 2019).

Wazuh

Wazuh es la plataforma de ciberseguridad de código abierto más usada para detección de riesgos, análisis de registros y cumplimiento de políticas de seguridad, se encarga de analizar datos de seguridad en endpoints, nubes y redes para localizar amenazas y responder a imprevistos, de este modo las organizaciones fortalezcan su seguridad mediante una monitorización continua (Wazuh, 2018).

TheHive

TheHive es una plataforma sólida dado que actúa como un centro de mando completo durante el análisis de un incidente. A demás cuenta con creación de usuarios y permite llevar cabo un seguimiento constante en tiempo real convirtiéndola en una aplicación significativa para centros SOC (Amine, 2023).

Cortex

Cortex es una plataforma de código abierto que tiene como objetivo el análisis automático de observables relacionados con incidentes de seguridad. Funcionan mediante analizadores que permiten consultar fuentes de información internas o externas con el fin de

obtener información sobre elementos como direcciones IP, dominios, URL, archivos y hashes, y mediante respondedores que permiten ejecutar acciones ya definidas sobre dichos elementos. Así, Cortex ayuda a aumentar la información técnica disponible durante la investigación de un incidente

MISP

MISP (Malware Information Sharing Platform) es una plataforma de Código abierto que favorece el intercambio de información relacionado con las amenazas cibernéticas y la colaboración entre las diferentes organizaciones. Su objetivo principal es fortalecer la capacidad de prevención, detección y respuesta de incidentes de seguridad a través de la recopilación y distribución de datos vinculados con amenazas (Centro de Gestión de Incidentes Informáticos, 2025).

Docker

Un Docker es una tecnología de código abierto destinada a la creación, implementación, ejecución y gestión de las aplicaciones en contenedores. Los contenedores se consideran componentes estandarizados de software que empaquetan código y sus dependencias necesarias para que se ejecute ese código de forma eficaz en varios entornos informáticos (Docker, 2018).

Docker-Compose

Docker Compose es una herramienta del ecosistema Docker creada para gestionar y ejecutar aplicaciones multicontenedor de manera conjunta. A diferencia de Docker que permite ejecutar contenedores individuales, Docker Compose facilita la orquestación entre contenedores, la administración de servicios, redes y volúmenes en un solo archivo de configuración YAML (Docker, 2018).

Indicador de Compromiso

Indicador de compromiso (IOC) hace referencia a datos o pistas que sugieren que un sistema o red pudo haber sido afectado por una posible intrusión o brecha de seguridad. Estos

indicadores facilitan la identificación, detección y respuesta ante incidentes de seguridad lo que hace posible una defensa proactiva (Fortinet, 2022).

Inteligencia de Amenazas

Inteligencia de amenazas (CTI en inglés) hace referencia al conocimiento obtenido y organizado sobre los riesgos y sus patrones de ataque. Esta información favorece a la comprensión del comportamiento de las amenazas para así poder anticipar posibles incidentes y apoyar la toma de decisiones (Kaspersky, 2024).

MTTD

MTTD o tiempo medio de detección es un indicador que se encarga de la medición del tiempo que tarda una entidad en descubrir un incidente. A partir de eso se generan umbrales reducidos para la detección temprana y la resolución inmediata de los problemas (SentinelOne, 2014).

MTTR

MTTR (Tiempo Medio de Respuesta) es un indicador que establece el intervalo promedio desde que existe una detección de un evento hasta el momento que se ejecuta un análisis o respuestas configuradas. Esta métrica valora la velocidad de funcionamiento del entorno desde la detección hasta la respuesta (Microsoft, 2025).

Capítulo III

Metodología

Enfoque de investigación

La investigación adoptó un enfoque tecnológico aplicado, basado en la elaboración de una solución funcional dentro de un ambiente específico. El proyecto se desarrolló en el laboratorio de Ciberseguridad, donde se creó un entorno para la coordinación, automatización y respuesta ante eventos de seguridad mediante herramientas específicas.

Este enfoque relacionó los elementos teóricos y técnicos orientados a la construcción de un ambiente comprobable. Su aplicación integró conocimientos sobre arquitectura, seguridad informática, contenedores e integración de servicios para atender una necesidad del laboratorio.

Alcance de la investigación

La presente investigación presentó un alcance exploratorio, debido a que se trató de un estudio inicial sobre plataformas SOAR en el laboratorio de Ciberseguridad. Los antecedentes indicaron referentes sobre este tipo de plataformas, mientras que el contexto académico brindó la oportunidad de analizar la aplicación conjunta de las herramientas seleccionadas. Este alcance facilitó la primera aproximación al objeto de estudio, la identificación de sus elementos principales y el establecimiento de bases para futuras investigaciones y optimizaciones dentro del laboratorio.

Además, la investigación presentó un alcance descriptivo, dado que caracterizó los elementos de la propuesta. La descripción incluyó las características, las funciones y conexiones dentro de la arquitectura general. Este alcance indicó la estructura del sistema y el papel asignado de cada componente durante su operación.

Método de investigación y desarrollo

La investigación utilizó Design Science Research Methodology (DSRM). Esta metodología se centra en la creación y evaluación de artefactos tecnológicos. Además, ofrece

una guía para el desarrollo de soluciones relacionadas con sistemas de información y organiza el proceso en pasos definidos (Muriyatmoko, Tjahyanto, Harmini, & Iqbal, 2024). En el presente estudio, el artefacto correspondió al entorno integrado para la detección, el análisis y la respuesta ante incidentes dentro del laboratorio.

La metodología comprende seis actividades: identificación del problema, definición de los objetivos, diseño y desarrollo, demostración, evaluación y comunicación. La identificación delimitó la necesidad atendida por el proyecto. La definición de los objetivos estableció los resultados esperados. El diseño y desarrollo comprendió la creación del entorno tecnológico. La demostración mostró cómo funcionaba en situaciones controladas. La evaluación analizó el desempeño con criterios técnicos. La comunicación recopiló el proceso, las pruebas y los resultados obtenidos (Muriyatmoko, Tjahyanto, Harmini, & Iqbal, 2024).

Diseño de investigación

La investigación adoptó un diseño cuasiexperimental, dado que estableció condiciones controladas para examinar el comportamiento del entorno. El escenario principal se evaluó en tres condiciones de carga: 5, 10 y 20 archivos, sobre la misma infraestructura tecnológica. Las variables consideradas fueron el tiempo medio de detección, el tiempo medio de respuesta, la clasificación de los eventos, el nivel de automatización y el consumo de recursos.

La evaluación incluyó dos escenarios complementarios: intentos fallidos de acceso mediante SSH y movimiento de un archivo protegido fuera de su ubicación autorizada. Ambos escenarios ampliaron la valoración funcional del entorno ante eventos de distinta naturaleza.

Población y muestra

Debido al carácter tecnológico del proyecto, la población fue de naturaleza técnica y abarcó los elementos que conformaron el entorno de ciberseguridad. Además, se consideró parte de la población los servicios, herramientas, plataformas, documentación, eventos creados y operaciones relacionadas con el rendimiento del sistema.

La muestra se seleccionó de forma intencional, debido al alcance técnico y a las funciones establecidas en la arquitectura. La Tabla 1 presenta los elementos incluidos en la muestra técnica y el rol asignado a cada uno.

Tabla 1

Componentes incluidos en la muestra técnica y su rol

Componente	Versión utilizada	Rol a desempeñar
Ubuntu Server	24.04.4 LTS	Sistema operativo de la máquina virtual donde se implementó el entorno
Docker Engine	29.6.1	Ejecución de los servicios contenerizados.
Docker Compose	5.3.0	Definición y administración del despliegue de los contenedores.
Wazuh Manager, Indexer y Dashboard	4.14.6	Detección, procesamiento, almacenamiento y visualización de las alertas.
Wazuh Agent 002	4.14.4	Monitoreo del equipo Ubuntu utilizado durante las pruebas.
TheHive	5.2	Gestión de alertas, casos y observables
Cortex	4.0.0-1	Análisis automático de los observables
MISP	2.4.177	Consulta de indicadores de compromiso e inteligencia de amenazas

Nota: Elaboración propia

Técnicas e instrumentos

La primera técnica fue la observación técnica, aplicada al registro de los acontecimientos producidos durante las pruebas, según las condiciones establecidas para cada ejecución. Esta técnica fue compatible con el proyecto por el aporte de información sobre errores, tiempos, evidencias y respuestas en tiempo real creadas por los componentes utilizados.

La segunda técnica corresponde a la revisión de los registros técnicos generados por los componentes del sistema. Estos registros aportaron información sobre las respuestas, los errores, los tiempos de ejecución y los resultados asociados a cada prueba efectuada. De este modo, la información obtenida se sustentó en evidencias producidas durante la operación de la plataforma.

Los instrumentos utilizados fueron las listas de verificación, logs del sistema, capturas de pantalla, resultados de los flujos y métricas de evaluación. Las listas de verificación confirmaron el estado y la configuración de los servicios, los logs sirvieron para revisar el comportamiento interno de los componentes. Las capturas sustentaron visualmente las pruebas, mientras que los registros y las tablas organizaron los datos obtenidos.

Validación, confiabilidad y consideraciones éticas

La validación de la lista de verificación se sustentó en criterios técnicos predefinidos. Estos criterios determinaron el grado de cumplimiento de las condiciones necesarias para la ejecución correcta de cada aspecto revisado dentro del entorno desarrollado.

La lista de verificación sirvió como instrumento de control para el registro del cumplimiento de cada criterio establecido. La confiablidad se respaldó mediante la aplicación consistente de los criterios definidos para cada escenario. Las repeticiones de una misma prueba conservaron las condiciones, la secuencia y los parámetros de evaluación establecidos. Los formatos de registro mantuvieron una estructura estable dentro de cada tipo de prueba, lo

cual facilitó la comparación de las evidencias correspondientes y preservó la coherencia de la información recopilada.

El proyecto se desarrolló mediante el uso responsable de los recursos tecnológicos disponibles en el laboratorio. Las actividades realizadas se limitaron al alcance definido para el Trabajo de Integración Curricular., con lo cual se previene cualquier acción que pueda afectar servicios externos, redes no autorizadas o sistemas que no formen parte del ambiente de pruebas. Las simulaciones utilizadas fueron con fines académicos y de validación. Por esta razón, el proyecto prescindió de datos reales de terceros y de información confidencial.

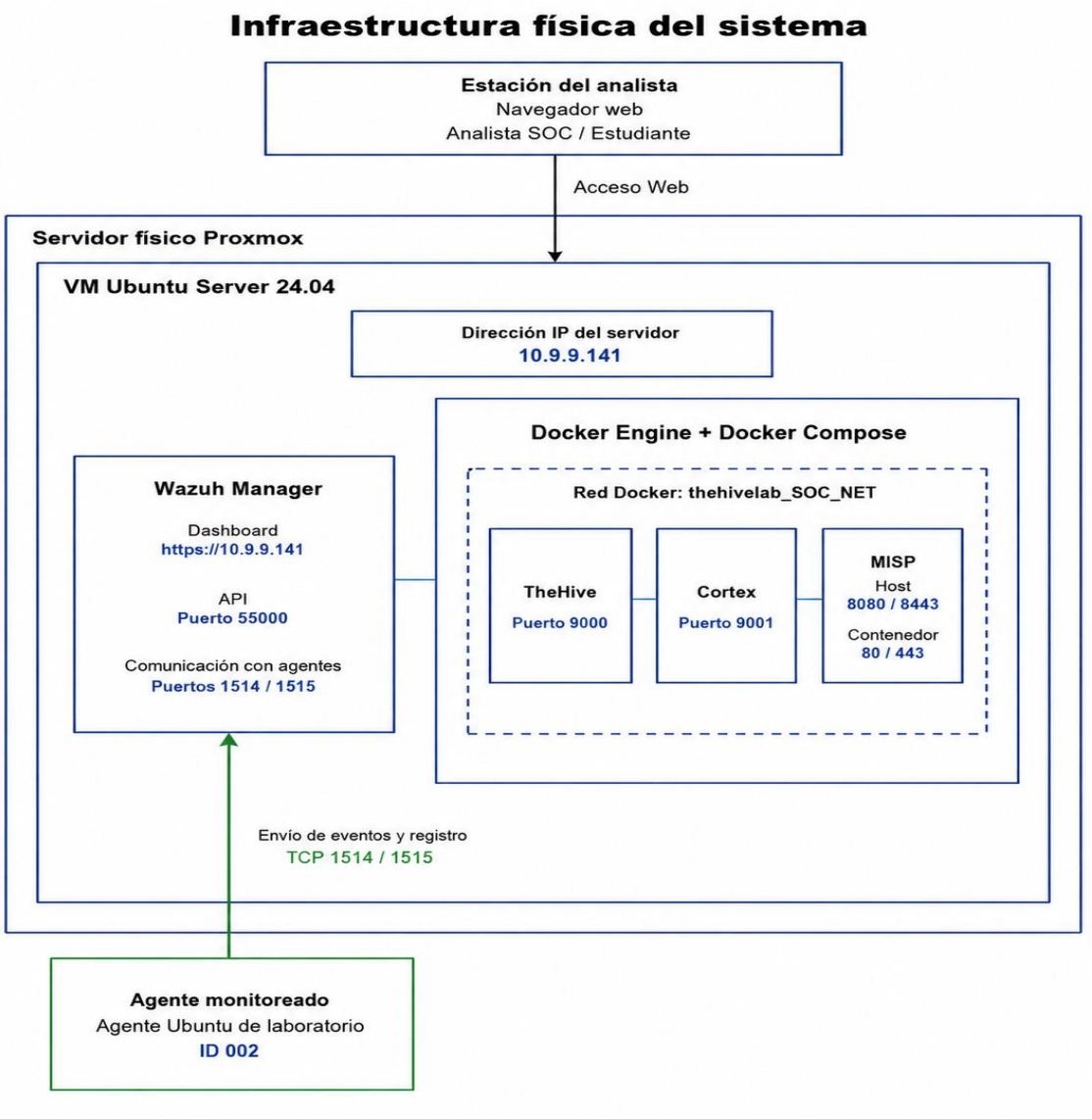
Las herramientas de ciberseguridad se usaron con fines formativos y técnicos, de acuerdo con las normas del laboratorio e indicaciones establecidas por el tutor a cargo. Esta delimitación garantizó la ejecución segura y controlada de las pruebas, de acuerdo con los objetivos académicos del proyecto.

Diseño físico de la arquitectura.

El diseño físico de la arquitectura presenta la distribución real de los elementos utilizados en la implementación del entorno. La infraestructura estuvo compuesta por un servidor físico Proxmox, una máquina virtual Ubuntu Server 24.04, Wazuh Manager, Docker Engine, Docker Compose, una estación de acceso para el analista y un agente Ubuntu monitoreado.

Figura 1

Infraestructura física del sistema



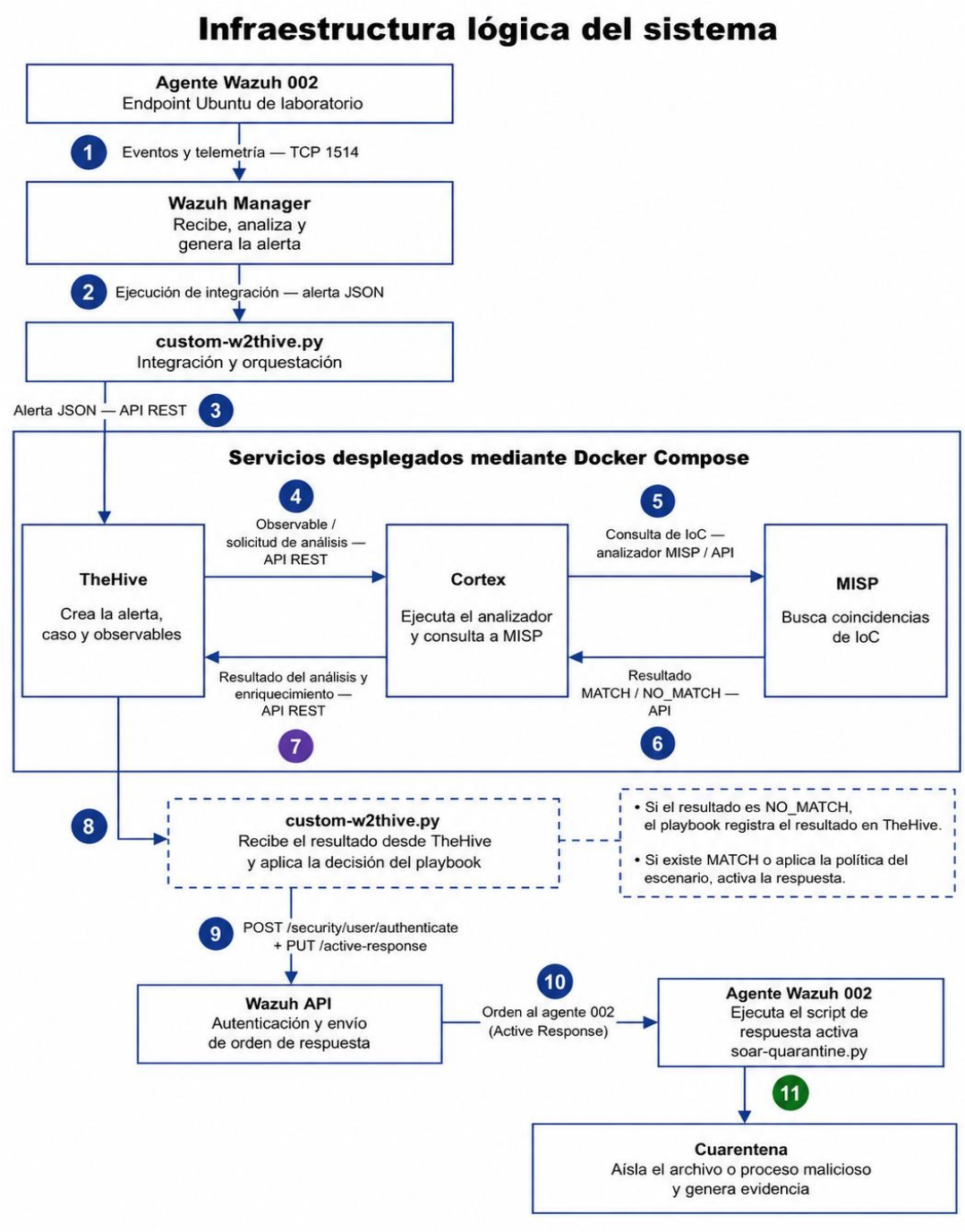
Nota: Elaboración Propia.

Diseño Lógico de la Arquitectura

El diseño lógico de la arquitectura presentó la secuencia funcional del sistema desde la creación de un suceso hasta la aplicación de una respuesta. La Figura 2 presenta la secuencia lógica de los componentes y el recorrido del evento dentro del entorno implementado

Figura 2

Diseño lógico del sistema



Nota: Elaboración Propia

Construcción del entorno base

La primera fase comprendió la organización del entorno base utilizado para la implementación de la plataforma. Dentro de esta fase se pudo verificó que el servidor del laboratorio tendrá los elementos necesarios para comenzar con el despliegue de los servicios de ciberseguridad. Las actividades se relacionan con la instalación de Wazuh Manager, la configuración principal de monitoreo, la instalación del agente y la configuración de Docker como tecnología de soporte.

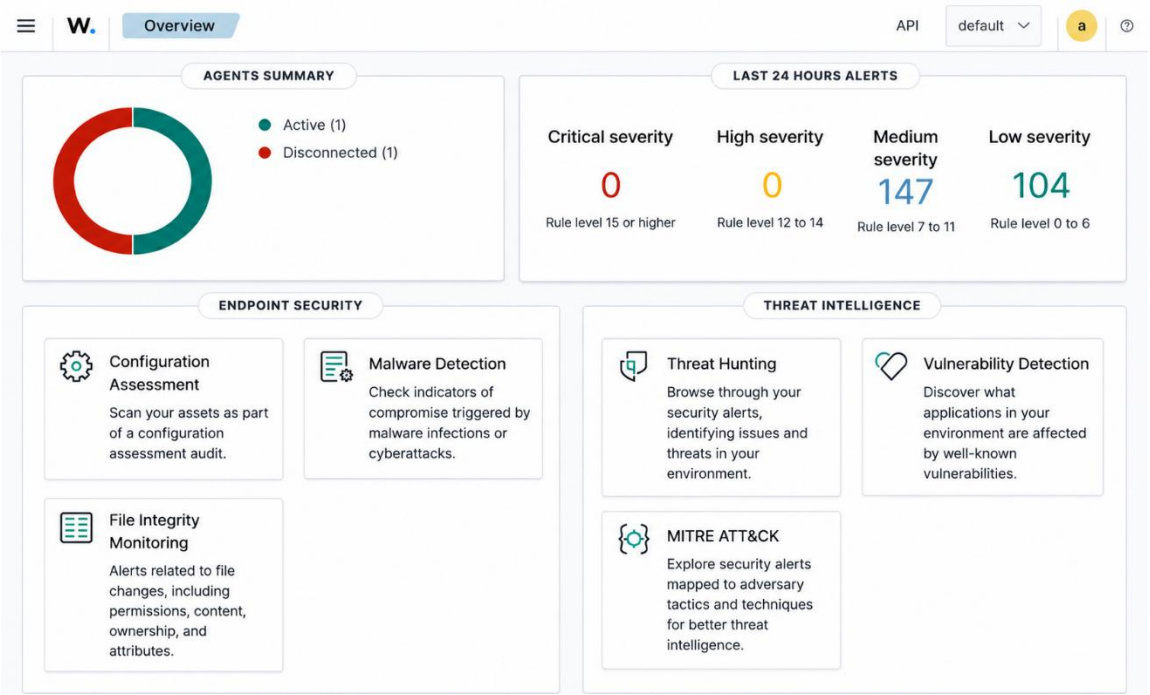
La implementación de Wazuh Manager proporcionó un componente central para la recepción, organización y visualización de eventos de seguridad. Según la evidencia de instalación se verifica que el proceso concluyó con éxito y que el sistema produjo las credenciales de acceso iniciales. El resultado fue fundamental para dar paso a las siguientes etapas, porque manager es un centro encargado de administrar la información que viene de los equipos monitoreados.

El acceso al Dashboard de Wazuh confirmó la disponibilidad de la interfaz web desde el navegador. Esto no solo representa una pantalla de acceso, sino más bien una comprobación de que los servicios fundamentales de manager estaban activos y que la plataforma facilitó la revisión de los eventos, alertas y estados del sistema.

La Figura 3 presenta el dashboard de Wazuh y el resumen del estado de los agentes y las alertas registradas en el entorno.

Figura 3

Dashboard de Wazuh levantado en el sistema.



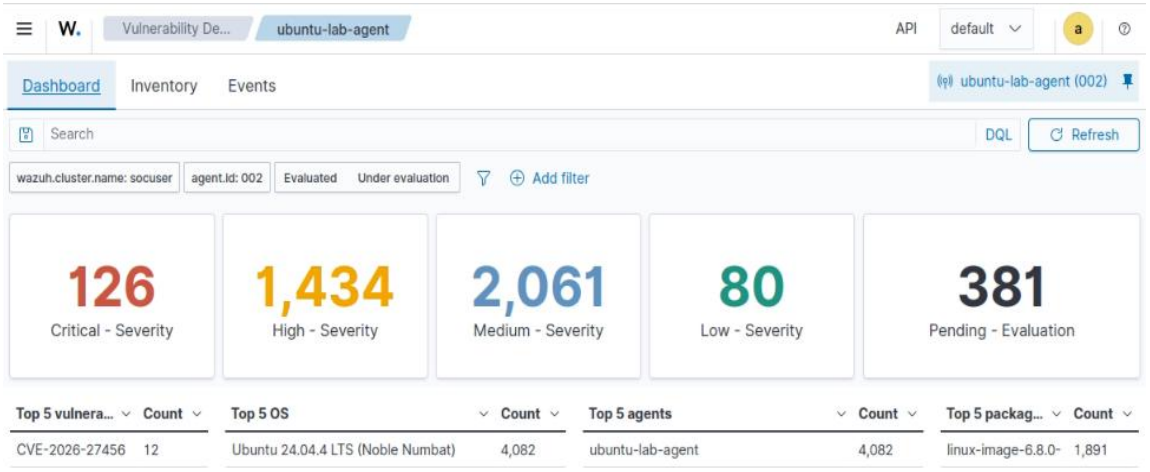
Nota: Pantalla principal de Wazuh

Después, se habilitó el reconocimiento de vulnerabilidades sobre el mismo servidor manager. Esta configuración fue clave porque en una instalación inicial, Wazuh no siempre busca analizar el equipo en donde se despliega. Al cambiar este parámetro y reiniciar el servicio, se mostró en el panel de control todas las vulnerabilidades clasificadas de acuerdo con su severidad. Este resultado muestra que el sistema aumentó su capacidad de monitoreo y ahora genera información sobre el estado de seguridad del servidor.

La visualización de vulnerabilidades en dashboard confirmó que Wazuh superó una instalación básica y aportó información técnica sobre los riesgos presentes en el entorno. Este resultado fue relevante para el proyecto, debido a que generó datos de seguridad aplicables en los flujos de análisis y respuesta. El detalle de las vulnerabilidades detectadas en los equipos monitoreados se muestra en la Figura 4 mediante el módulo correspondiente de Wazuh.

Figura 4

Vulnerabilidades analizadas en Wazuh Dashboard

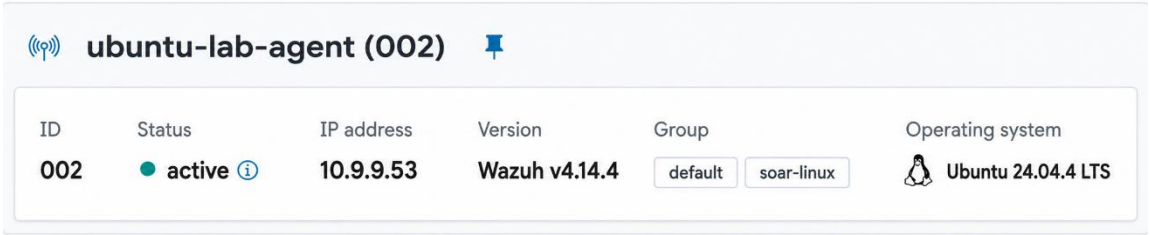


Nota: Interfaz con las vulnerabilidades detectadas por Wazuh.

Adicional, se implementó el agente de Wazuh sobre un sistema Linux Ubuntu Server. Esta incorporación añadió un endpoint al proceso de monitoreo y permitió la generación de eventos desde un equipo supervisado. La existencia del agente agrega la función del manager para obtener información desde un sistema monitoreado y mandarla hacia el elemento central para su procesamiento. La incorporación del agente Ubuntu al proceso de monitoreo se observa en la Figura 5, junto con la información registrada por Wazuh.

Figura 5

Agente Ubuntu monitoreado en Wazuh.



Nota: Agente Ubuntu monitoreado y activo en Wazuh.

Dentro de esta misma etapa, se instaló Docker y Docker Compose. El estado activo de Docker confirmó la capacidad del servidor para ejecutar contenedores. Además, se comprobó que la infraestructura base se encontraba preparada para la implementación de servicios múltiples mediante archivos de configuración.

Los resultados de esta primera etapa muestran que el entorno base quedó configurado para soportar una arquitectura de ciberseguridad integrada. Wazuh quedó habilitado como fuente de eventos y Docker como un soporte para la realización de servicios en contenedores. Este conjunto sustentó el posterior despliegue de las plataformas encargadas de la gestión de casos y del análisis automatizado.

Despliegue de servicios contenerizados

Después de que el servidor está listo, se empezó con el despliegue de los diferentes servicios que forman parte de la arquitectura de soporte a la plataforma. La utilización de contenedores ayudó a ejecutar diferentes componentes dentro de una misma infraestructura, con una separación lógica entre servicios, además de simplificar su administración a través de Docker Compose.

La creación del directorio de trabajo ayudó a organizar archivos necesarios para el despliegue. Para esta estructura se situaron configuraciones, carpetas de soporte y elementos requeridos por los servicios. El resultado indica que la implementación no solo se realizó de forma aislada, sino con el propósito de centralizar los elementos técnicos de entorno.

Previo al levantamiento de los contenedores se hizo ajustes específicos en el sistema anfitrión. La configuración del parámetro “vm.max_map_count” fue uno de ellos, exigido por Elasticsearch para su correcta operación. Este cambio evidencia que el despliegue de una arquitectura en contenedores no solo depende de ejecutar servicios, sino de preparar un sistema base que satisfaga los requisitos de componentes que dirigen el almacenamiento, búsqueda e indexación de información.

Por otro lado, se crearon carpetas orientadas a configuraciones, archivos, registros y trabajos temporales. Este resultado es relevante porque algunos servicios necesitan rutas persistentes para mantener datos o almacenar la información generada en el transcurso de su funcionamiento. Para el caso de Cortex, el archivo “application.conf” se configuró, para lo cual se definieron diferentes lineamientos relacionados con el puerto de servicio, directorios internos y clave de aplicación.

La configuración de Cortex posibilitó preparar el elemento encargado de la evaluación automática de observables. Este resultado es porque Cortex no hace lo mismo que TheHive o MISP; sino su función está orientada al procesamiento de indicadores técnicos a través de analizadores y respondedores. Por esto, como elemento necesario se encuentra la disponibilidad dentro del stack para que el entorno ejecute acciones de análisis sobre componentes sospechosos.

Luego de establecer la dirección IP del servidor en el archivo “docker-compose.yml”, se realizó el levantamiento de los servidores. Los servicios que se pudieron identificar son TheHive, Cortex, MISP, Elasticsearch y otros módulos asociados. El resultado indica que la arquitectura logró desplegarse de forma simultánea a través de Docker Compose. La disponibilidad simultánea de los servicios desplegados se evidencia en la Figura 6 mediante el listado de contenedores activos.

Figura 6

Contenedores del entorno encendidos.

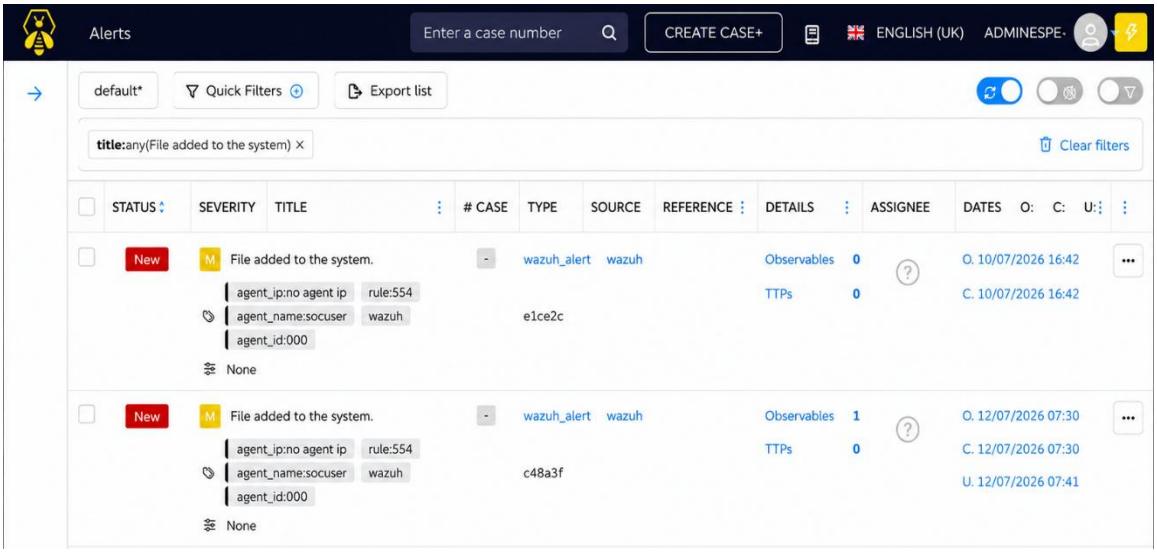


Nota: Contenedores encendidos mediante el uso de Docker Compose

La habilitación de las interfaces web hizo posible comprobar que desde el navegador las plataformas principales quedaron accesibles. Para la administración de alertas y casos se encuentra TheHive, para el análisis de observables Cortex, para la gestión de inteligencia de amenazas MISP. Toda esta verificación permitió comprobar que los servicios quedaron operativos para avanzar con la integración funcional. El acceso a las funciones de gestión de alertas y casos se muestra en la Figura 7 mediante la interfaz disponible de TheHive.

Figura 7

Interfaces web de TheHive

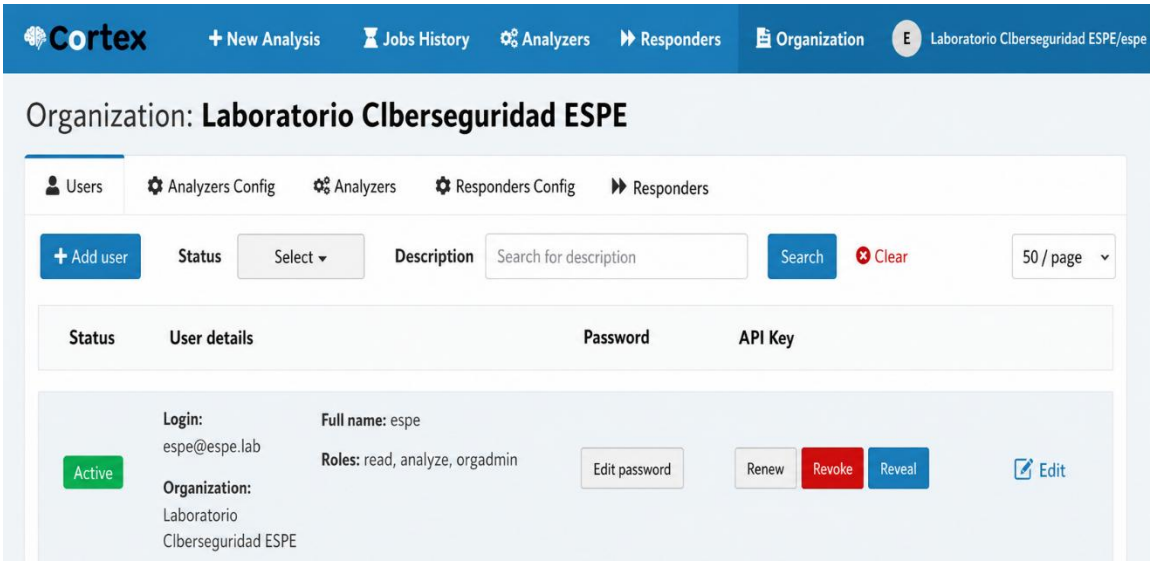


Nota: Pantalla principal de TheHive

A continuación, se muestra en la Figura 8 la configuración aplicada para la conexión de Cortex dentro del entorno implementado.

Figura 8

Interfaces web de Cortex

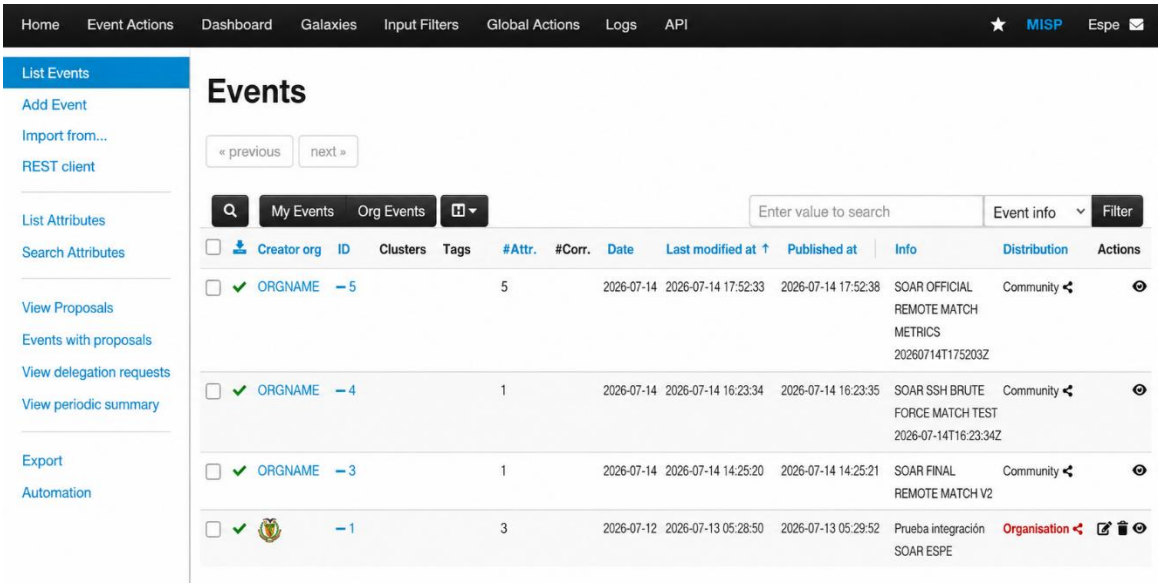


Nota: Interfaz web de Cortex dentro la organización creada para el laboratorio.

En la figura 9 se observa la disponibilidad de MISP para la gestión y consulta de información sobre amenazas

Figura 9

Interfaces web de MISP



Nota: Interfaz web de MISP con la organización creada para el laboratorio.

El resultado obtenido de la implementación del entorno basado en contenedores contribuye al objetivo principal del proyecto, porque permite contar con una arquitectura conformada por varios servicios especializados. Al contrario de una instalación individual de herramientas, esto permite que cada elemento tenga una función específica en el flujo de seguridad: detección, gestión, análisis y almacenamiento.

Integración entre TheHive, Cortex y MISP

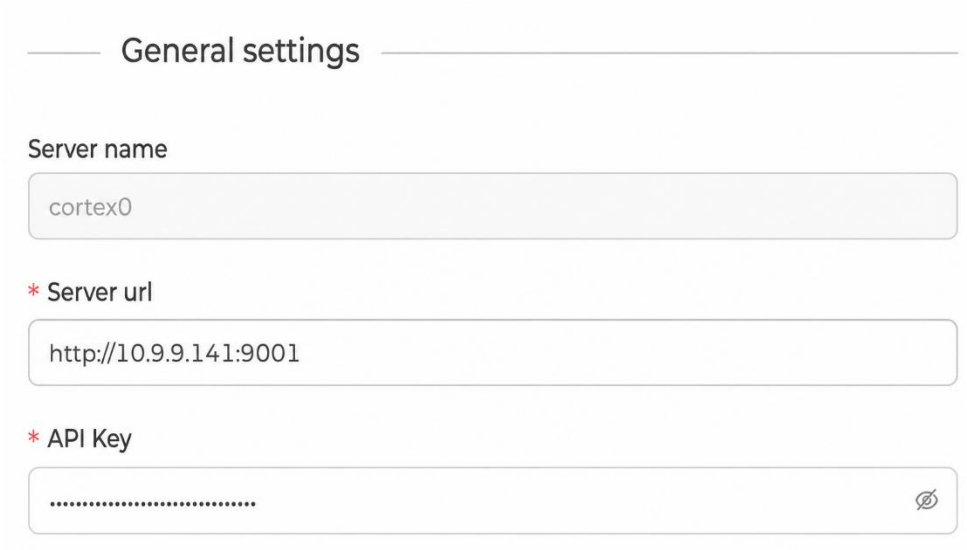
Después del despliegue de los servicios, se ejecutó el ajuste funcional entre TheHive, Cortex y MISP. En esta fase se logró transformar de una arquitectura compuesta de herramientas disponibles de forma individual a un entorno comunicado entre plataformas. La integración se consiguió a través de la creación de organizaciones, usuarios y claves API, componentes esenciales para habilitar el intercambio de información entre los servicios.

Se configuró un esquema de trabajo asignada al laboratorio en TheHive. Esto permitió que el sitio donde serían recibidos y administrados los eventos de seguridad obtenidos en las pruebas se delimiten. El desarrollo de usuarios asociados a la organización facilitó la elaboración del acceso operativo a la plataforma y determinar una base para el manejo de alertas y casos.

La configuración de Cortex en TheHive facilitó el análisis automatizado de observables. Por esta razón, se originó una clave API desde Cortex y se evidenció la conexión en TheHive. Es importante tener en cuenta este resultado porque permite que TheHive no solo reciba alertas, sino más bien pueda apoyarse en Cortex para realizar análisis técnicos sobre indicadores como direcciones IP, archivos, URL asociado con una alerta. La vinculación establecida entre TheHive y Cortex se evidencia en la Figura 10 mediante los parámetros definidos para el intercambio de información.

Figura 10

Conexión de Cortex configurada en TheHive.



Nota: Configuración de Cortex dentro de TheHive.

Para el punto de vista funcional, la incorporación con Cortex figura una mejora en el proceso de investigación porque disminuye la necesidad de verificar de forma manual cada observable mediante fuentes externas. En su lugar, el criterio del analista se complementa con resultados técnicos generados por los analizadores lo que fortalece la interpretación del evento.

Como adicional se agregó de MISP dentro del ciclo de análisis como la principal fuente de inteligencia de amenazas. Esta integración ayudó a que los observables analizados no depende solo de un chequeo interno, sino que logren ser minimizados con indicadores registrados en la plataforma centralizada. Con esto se aumenta un marco de contexto para reforzar la explicación de los eventos examinados desde TheHive.

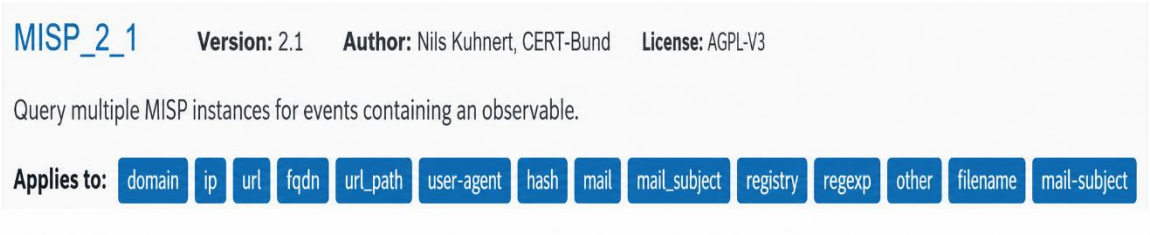
Para establecer esta conexión se crearon organizaciones correspondientes, usuarios dentro de las mismas y claves API dentro de las aplicaciones participantes. Estas configuraciones aportaron para establecer permisos y procedimientos de autenticación entre las plataformas, de modo que las acciones dependen de accesos manuales. Así se logró la

comunicación entre TheHive a Cortex para solicitar análisis y Cortex se estableció para extraer información almacenada en MISP.

La incorporación de MISP en Cortex amplió la función de los analizadores establecidos. Para la práctica se declaró el analizador MISP_2_1 creado para consultar indicadores registrados en MISP y relacionarlos con los observables mandados desde TheHive. Esto se hizo para enlazar el análisis automático con una fuente de inteligencia de amenazas estructurada. La disponibilidad del analizador MISP dentro de Cortex se presenta en la Figura 11, como parte del proceso de consulta de indicadores asociados con los observables.

Figura 11

Analizador MISP habilitado en Cortex.



Nota: Analizador MISP habilitado en Cortex.

Para entrar en la parte de la validación, en la plataforma MISP se crearon eventos controlados y se difundieron con atributos técnicos asociados. Direcciones IP y hashes se agregaron dentro de los atributos, debido a que estos factores cumplen un funcionamiento como indicadores de compromiso en un escenario de verificación. Debido a la publicación de dichos eventos la información queda a la mano para ser consultada durante el análisis automático.

Cabe mencionar que este resultado permitió comprobar que MISP funciona como una fuente de referencia en el flujo de análisis creado. A partir de los indicadores publicados, la plataforma Cortex puede consultar información disponible y entregar una respuesta relacionada

con el observable enviado desde TheHive. De tal manera, se demuestra que no solo existe una conexión entre plataformas, sino que, MISP actúa como repositorio de los indicadores durante la revisión de alertas y casos. Esta relación hace que los observables no sean manejados como datos aislados sino como elementos comparables con información de inteligencia almacenada.

Mientras se realizaba validaciones de la interconexión del sistema, se presentó resultados de clase MATCH y NO_MATCH. El tipo MATCH indica que el observable evaluado tiene relación con un atributo anotado en MISP, mientras, que el tipo NO_MATCH indica que no se encontró ninguna coincidencia a la hora de la consulta. Estas respuestas son fundamentales para comprobar que el analizador no solo realiza una consulta automática sin sentido, sino que brinda una respuesta resolutiva para orientar la verificación del indicador.

Cabe mencionar que al obtener un resultado de tipo NO_MATCH no confirma que el observable sea seguro, sino que dentro de la información registrada en MISP no fue encontrado. En cambio, una respuesta MATCH posibilita el aumento de la relevancia del observable dentro del análisis, debido a que lo vincula con información de inteligencia registrada. Es por eso que, al presentarse una coincidencia, el observable puede ser marcado como IOC (Indicadores de compromiso) en TheHive, de este modo se logre una fácil identificación durante la revisión de la alerta o caso correspondiente.

Integración entre Wazuh y TheHive

Una vez realizada la comunicación entre las plataformas levantadas en Docker, se continuó con la conexión entre Wazuh y TheHive. Debido a esta unión se logró conectar la fuente de generación de alertas con la plataforma responsable de recibir y ordenar los eventos de seguridad. Así Wazuh se estableció como un elemento de detección y TheHive como un componente de recepción para comenzar con el análisis correspondiente de los eventos.

La mencionada conexión se estableció mediante dos archivos “custom-w2thive.py” y “custom-w2thive”, los mismos ubicados dentro de la carpeta de integraciones de Wazuh. Estos

dos archivos permiten tomar las alertas generadas por Wazuh y mandarlas hacia TheHive a través de una comunicación establecido por API. De forma inmediata se designaron permisos y propietario a los archivos correspondientes, esto con el fin de que el proceso se pueda realizar de manera correcta, lo que permitió que Wazuh Manager ejecute la integración cuando una alerta cumpla con las condiciones configuradas.

A continuación, se modificó el archivo “ossec.conf” para aumentar el módulo de integración. En la configuración se especificó el nombre del script, la dirección correspondiente del servidor TheHive, la API Key y el modelo de alerta en JSON. A través del formato establecido se maneja una estructura ordenada de la información enviada, al agrupar datos como el nivel de severidad, la regla activada, el agente y una descripción esencial del evento. La configuración que permitió el envío de alertas desde Wazuh hacia TheHive se muestra en la Figura 12 mediante los parámetros establecidos para la integración.

Figura 12

Configuración de integración entre Wazuh y TheHive.

```
root@socuser:/home/socuser/evidencias-soar# sudo grep -n -A8 -B2 "custom-w2thive" /var/ossec/etc/ossec.conf
81-  </wodle>
82-<integration>
83:    <name>custom-w2thive</name>
84-    <level>5</level>
85-    <hook_url>http://10.9.9.141:9000</hook_url>
86-    <api_key>w4IUKLvtS0cnSSDf4p6fUD0IKUwrKn4L</api_key>
87-    <alert_format>json</alert_format>
88-  </integration>
```

Nota: Bloque de configuración custom-w2thive dentro del archivo ossec.conf..

De igual forma se determinó un umbral para notificar alertas con un nivel de severidad definido. Esto con el fin de reducir el envío de eventos de baja relevancia y dar mayor importancia a alertas con más importancia operativa. Esto hace que TheHive obtenga información más concreta y evite tener exceso de registros que pudieran dificultar el análisis técnico de cada alerta.

Configuración de respuesta activa y cuarentena

Una vez terminado la integración de todas las plataformas, se comprobó la capacidad del sistema frente a un evento detectado con el fin de ejecutar una acción de contención. Este punto es clave, ya que, permitió analizar si el sistema lograba pasar desde la creación de una alerta a la ejecución de una medida técnica sobre el elemento implicado. Para ellos, se utilizó de Wazuh con su mecanismo de respuesta en un escenario controlado de laboratorio.

Con la ayuda del comando “soar-quarantine” y el script “soar-quarantine.py” se configuró una acción de respuesta. Dicha línea registrar la acción dentro de Wazuh, y el script ejecuta el proceso de cuarentena sobre el archivo identificado para evitar la afectación al sistema. De tal forma los dos elementos trabajan de forma simultánea, uno como definición de la respuesta y el otro como encargado de aplicarla.

Todo el procedimiento mencionado se llevó a cabo en una ruta controlado, para así, evitar acciones fuera del alcance establecido. La carpeta “/var/ossec/soar-incoming” fue definida como directorio monitoreado, en cambio “/var/ossec/quarantine” se definió como el destino para los archivos aislados. Debido a esta separación se puede comprobar si el archivo es pasado del directorio inicial al espacio destinado para su revisión posterior.

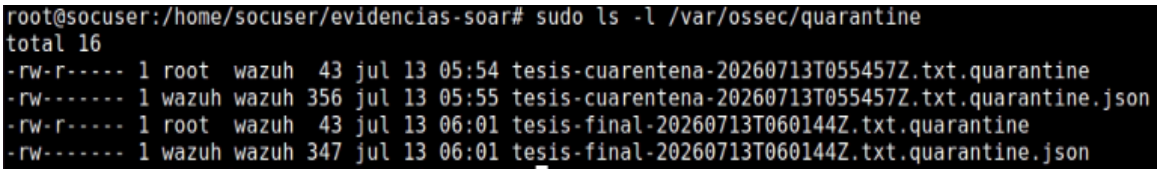
La respuesta fue ejecutada sobre el agente de forma remota a través de la API de Wazuh. Esta respuesta mostró que la acción puede ser enviada desde el operador central hacia un endpoint gestionado, sin tener la necesidad de interactuar con el equipo monitoreado. Eso comprueba la vinculación entre la administración central con la ejecución técnica sobre un agente específico.

Como respaldo para este proceso, tras la ejecución de la respuesta se genera la evidencia correspondiente en formato JSON. Esto permite identificar datos asociados a la acción aplicada, cual es el archivo en proceso y el resultado obtenido. A demás, la regla de confirmación permite reflejar dentro del monitoreo que la ejecución fue reconocida por el

sistema y tuvo una implementación exitosa. La aplicación de la respuesta activa y el traslado del archivo hacia el directorio de cuarentena se evidencian en la Figura 13.

Figura 13

Evidencia de cuarentena generada por la respuesta activa.



Nota: La figura muestra los archivos trasladados hacia la ruta /var/ossec/quarantine y la evidencia JSON generada como respaldo de la respuesta activa ejecutada por Wazuh.

Configuración de escenarios automatizados

Una vez comprobada la integración entre los elementos principales y la obtención de una respuesta activa, El conjunto de archivos marcados como sospechosos fue la prueba principal. Esto permitió ver cómo funcionaba la clasificación, cuánto tardaba en detectar problemas, cuánto tiempo se necesitaba para responder y cuántos recursos se usaban en distintas situaciones. Por otro lado, los escenarios de ataque por fuerza bruta en SSH y movimiento no autorizado de un archivo protegido se usaron como pruebas adicionales.

A continuación, en la Tabla 2 se presentan los escenarios definidos para la validación del entorno implementado.

Tabla 2

Escenarios automatizados definidos para la validación del entorno

Escenario	Condición Simulada	Herramientas	Evidencia esperada
Carga de Archivos sospechosos	Archivo controlado cuyo hash está registrado como amenaza	Wazuh, TheHive, Cortex y MISP	Alerta y caso creado, amenaza confirmada y archivo en cuarentena

Fuerza bruta SSH	Múltiples intentos fallidos de acceso desde una misma dirección IP	Wazuh, TheHive, Cortex y MISP	Alerta generada, dirección IP analizada y caso registrado
Movimiento no autorizado de un archivo protegido	Traslado controlado de un PDF simulado desde la carpeta autorizada hacia una ubicación no permitida	Wazuh, TheHive, Cortex y MISP	Alerta y caso creado, análisis del archivo y traslado a cuarentena como medida de contención

Nota: Elaboración Propia

Escenario de detección de archivos ante distintos niveles de carga

El escenario principal consistió en agregar archivos de prueba al directorio supervisado del agente Ubuntu con ID 002. Los archivos no tenían información relevante ni código malicioso, por lo tanto, no se comprometió la seguridad del laboratorio durante la prueba realizada.

Se generaron 35 archivos con valores SHA-256 diferentes. De estos, 14 hashes se encontraban registrados en MISP para representar eventos con coincidencia. Los 21 restantes no tenían registro alguno y fueron empleados para evaluar el flujo sin coincidencia. Los archivos se clasificaron en tres categorías: carga baja (5 archivos), carga media (10 archivos) y carga alta (20 archivos).

En cada ejecución, los archivos se agregaron casi al mismo tiempo en la ubicación vigilada. Wazuh detectó los cambios y generó alertas. Luego, los hashes se procesaron y se consultaron en MISP. Si había una coincidencia, el archivo se movió a cuarentena. Si no había coincidencia, el evento se guardó y el proceso terminaba sin tomar esa acción.

Este escenario sirvió para evaluar la clasificación de los 35 archivos, el tiempo promedio de detección, el tiempo promedio de respuesta de los 14 eventos con coincidencia y el uso de CPU y memoria en cada nivel de carga.

Escenario de fuerza bruta SSH

El escenario a continuación se basa en la simulación de varios intentos fallidos de autenticación mediante una conexión SSH, esta prueba permite simular múltiples accesos no autorizados sobre el sistema monitoreado. Esto con el fin de comprobar que la herramienta principal identifique la repetición de intentos fallidos y pueda generar una alerta de este extraño comportamiento.

Este escenario se preparó con ocho intentos de autenticación con un usuario no existente. Esto se hizo en un lapso de 120 segundos. Todos estos intentos se realizaron desde la dirección IP 198.51.100.77. Esta dirección ya estaba registrada en MISP, lo que nos permitió comprobar si era un indicador de algo. No se bloquea de forma automática esta dirección porque se perdería el acceso al entorno del laboratorio.

Escenario de movimiento no autorizado de un archivo protegido.

Por último, como escenario complementario, se plantea la simulación del movimiento no autorizado de un archivo protegido dentro del entorno monitoreado, donde el servidor es considerado una infraestructura que posteriormente podrá destinarse al almacenamiento de documentación interna del laboratorio como manuales, procedimientos y guías operativas.

Se utilizó un archivo PDF de prueba con este tipo de documentación, pero sin información sensible ni datos reales del laboratorio. Se depositó en primer lugar en una carpeta determinada como ubicación autorizada y posteriormente fue trasladado de forma controlada a una carpeta no autorizada en el mismo entorno. Con esta acción se pudo representar un

incumplimiento de una política de seguridad que establece que los archivos protegidos no pueden ser movidos o copiados sin autorización.

Los valores SHA-256, SHA-1 y MD5 del documento no estaban registrados previamente en MISP. Esto se debió a que el objetivo no era detectar malware, sino verificar la detección de un comportamiento no autorizado. Wazuh tenía que reconocer el cambio de ubicación, generar una alerta y enviar los datos a TheHive. Luego, Cortex debía analizar los hashes y consultar a MISP para obtener más información sobre el archivo.

Como respuesta el documento debe trasladarse temporalmente a cuarentena porque no se cumplió la política de ubicación. Esta acción se tomó para evitar que se hicieran nuevas operaciones sobre el archivo y preservar la evidencia. Un analista determina si el movimiento del documento fue autorizado. La prueba también valida la lógica que se va a aplicar después sobre la ubicación definitiva del repositorio documental del laboratorio.

Capítulo IV

Análisis de los resultados

Este capítulo hace referencia al estudio de los resultados alcanzados en la implementación y verificación del entorno implementado para la automatización de detección, análisis y respuesta ante los incidentes de seguridad que se presenten. En contraste con los anteriores capítulos, no solo se busca describir una base teórica, sino consiste en interpretar los resultados obtenidos mediante las evidencias generadas en el desarrollo práctico del proyecto.

Las capturas, registros técnicos, estados de servicios y pruebas controladas en el ambiente designado forman parte de este análisis. Se revisa cada evidencia en base al papel que cumple dentro de la arquitectura puesta en práctica, de acuerdo con la disponibilidad de los servicios y la capacidad del entorno para analizar eventos de seguridad.

Resultados Generales de los escenarios evaluados

La evaluación comprendió tres escenarios construidos, cuyos resultados generales se indican en Tabla 3.

Tabla 3

Resultados generales de los escenarios evaluados.

Escenario	Resultados del análisis	Respuesta Obtenida	Estado
Carga de Archivos Sospechosos	Coincidencia del hash con un indicador registrado	Generación del caso y cuarentena del archivo	Exitoso
Fuerza bruta SSH	Identificación de accesos fallidos reiterados	Generación del caso y registro de la dirección IP	Exitoso

Movimiento no autorizado de un archivo	Ausencia de coincidencias en los hashes evaluados	Generación del caso y cuarentena por política	Exitoso

Nota: Elaboración Propia

La principal diferencia se determina en la acción de respuesta que se obtuvo en cada escenario. Los contextos relacionados con archivos sospechosos o con una posible extracción de información concluyeron con el traslado de los mismos a cuarentena, con el fin de evitar afectaciones al sistema, mientras que los intentos de conexión masivos por SSH se configuró para conservar su registro y análisis de la dirección IP. Esto evidencia que no se implementó las acciones de respuesta de forma uniforme, sino que se aplicaron decisiones diferentes para cada evento.

Resultados del escenario de carga de archivos sospechosos.

En este escenario se llevó a cabo en tres niveles de carga, en la Tabla 4 se indica la cantidad de archivos que se usó para cada medición, tratándose de archivos con coincidencia y sin coincidencia procesados en cada nivel.

Tabla 4

Distribución de archivos sospechosos en cada nivel

Nivel de Carga	Archivos con Coincidencia	Archivos sin Coincidencia	Total
Baja	2	3	5
Media	4	6	10

Alta	8	12	20
Total	14	21	35

Nota: Elaboración Propia.

En la tabla se muestra los 3 niveles de carga. La carga baja se compuso de 5 archivos, la carga media de 10 y la carga alta de 20. De este modo, la cantidad de archivos se duplicó con cada nivel, hasta llegar a un total de 35 elementos evaluados.

Dentro de la carga baja se utilizaron 2 archivos con coincidencia y 3 sin coincidencia, en la carga media 4 y 6, mientras que en la carga alta se evaluaron 8 y 12, respectivamente. En total hubo 14 archivos con coincidencia y 21 que finalizaron sin coincidencia.

Esta división representa los dos flujos que se han considerado en el escenario. Los archivos coincidentes se corresponden con el flujo que identifica un indicador registrado en MISP y ejecuta la respuesta de cuarentena. Por otra parte, los archivos sin coincidencia representan el flujo en que el análisis finaliza sin aplicar la acción mencionada.

Resultados del tiempo medio de detección

Para este apartado se tuvo en cuenta los 35 archivos de los tres niveles de carga, se tomó en cuenta desde integración de cada archivo en el directorio monitoreado hasta la generación de la alerta por Wazuh. A continuación, se muestra la Tabla 5 con el tiempo de detección en cada nivel.

Tabla 5

Tiempo medio de detección según los niveles de carga.

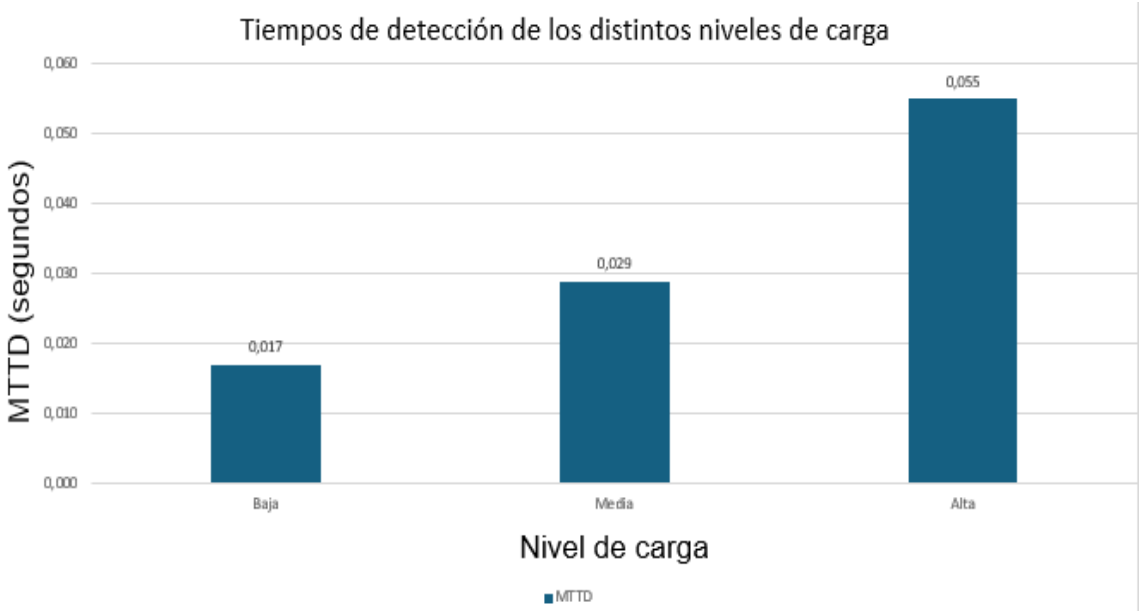
Nivel de carga	Archivos evaluados	Tiempo de detección (s)
Baja	5	0,017
Media	10	0,029
Alta	20	0.055
Total	35	0.042

Nota: Elaboración Propia

La variación del tiempo medio de detección entre los tres niveles de carga se representa en la Figura 14.

Figura 14

Comparación del tiempo de detección de los distintos niveles de carga



Nota: Elaboración Propia

Los resultados muestran que en la carga baja se obtuvo un promedio de 0,017 segundos tras procesarse 5 archivos, en la carga media se alcanzó 0,029 segundos con 10 archivos y en la carga alta con 20 archivos se registró 0,055 segundos. De forma general, el tiempo medio de detección de los 35 archivos fue de 0,042 segundos.

Resultados del tiempo medio de respuesta

Los tiempos medios de respuesta obtenidos en cada nivel de carga se presentan en la Tabla 6, junto con la cantidad de archivos procesados en cada condición.

Tabla 6

Resultados del tiempo de respuesta según el nivel de carga.

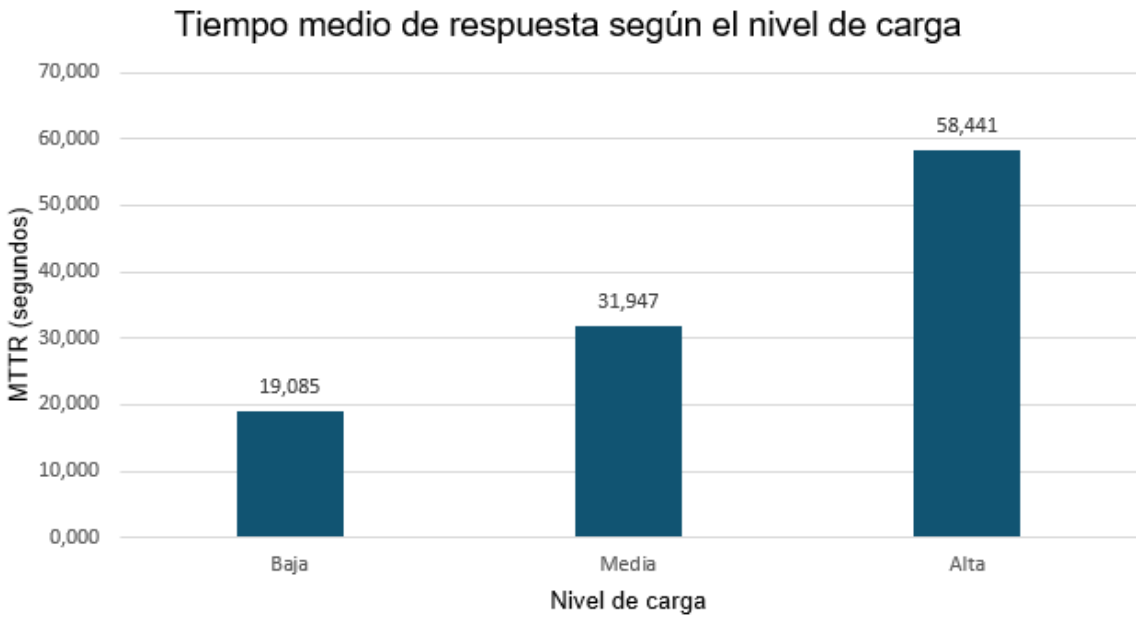
Nivel de carga	Archivos evaluados	Tiempo de respuesta (s)
Baja	2	19,085
Media	4	31,947
Alta	8	58,441
Total	14	45,249

Nota: Elaboración Propia

La representación gráfica de los tiempos medios de respuesta obtenidos en cada nivel de carga se presenta en la Figura 15.

Figura 15

Comparación del tiempo de respuesta de los distintos niveles de carga.



Nota: Elaboración Propia

Los resultados indican que el tiempo que se tarda en responder es mayor cuando hay más archivos que coinciden. Cuando hay 2 archivos, el tiempo medio de respuesta es de 19,085 segundos. Con 4 archivos el tiempo medio de respuesta sube a 31,947 segundos. Y con 8 archivos, el tiempo medio de respuesta llega a 58,441 segundos. De manera general, el tiempo medio de respuesta de los 14 archivos fue de 45,249 segundos.

Resultados del nivel de automatización.

Las actividades consideradas para valorar el nivel de automatización del flujo se detallan en la Tabla 7, de acuerdo con las etapas ejecutadas en cada escenario. Se estableció como automática aquella actividad que se ejecuta por el entorno sin una intervención manual.

Tabla 7

Actividades consideradas para evaluar el nivel de automatización.

Actividad evaluada	Forma de Ejecución
Detección del evento	Automática
Extracción del observable	Automática
Generación de la alerta	Automática
Consulta y análisis del indicador	Automática
Clasificación del resultado	Automática
Creación del caso	Automática
Ejecución de la acción definida según la clasificación	Automática
Registro de la evidencia	Automática

Nota: Elaboración Propia

El porcentaje de actividades completadas de forma automática dentro del flujo evaluado se observa en la Figura 16.

Figura 16

Porcentajes de actividades ejecutadas de forma automática.



Nota: Elaboración Propia

El resultado obtenido indica que una vez se generó el evento, el entorno llevó a cabo todas las acciones que son la extracción del observable, generación de la alerta, consulta y análisis del indicador, clasificación del resultado, creación del caso, ejecución de la acción definida y registro de evidencia determinadas sin necesitar del usuario hiciera el cambio de información entre plataformas, inicie los análisis o ejecute una acción definida.

Resultados de clasificación.

Los resultados obtenidos en la clasificación de los eventos se presentan en la Tabla 8, según las categorías definidas para valorar la efectividad de la detección.

Tabla 8

Resultados de las pruebas de efectividad de la detección.

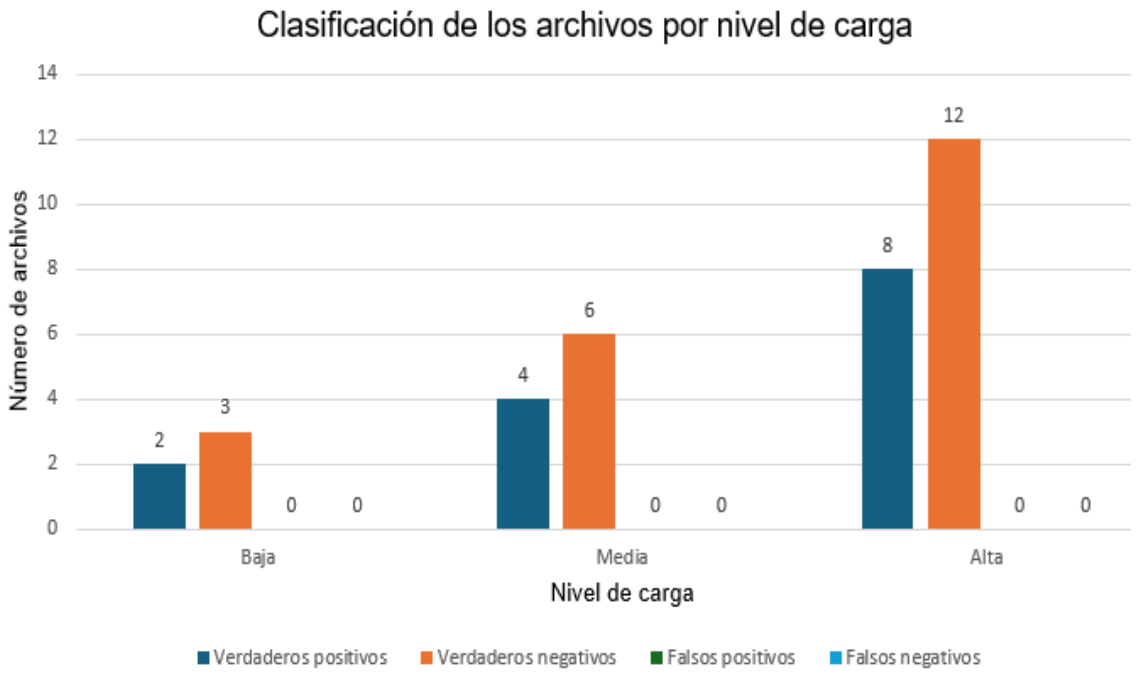
Nivel de Carga	Verdaderos Positivos	Verdaderos Negativos	Falsos Positivos	Falsos Negativos
Baja	2	3	0	0
Media	4	6	0	0
Alta	8	12	0	0
Total	14	21	0	0

Nota: Elaboración Propia

La distribución de los resultados según las categorías establecidas para la clasificación se representa en la Figura 17

Figura 17

Distribución de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos



Nota: Elaboración Propia

Los resultados de los 35 archivos procesados fueron clasificados de forma correcta. En la carga baja se determinó 2 verdaderos positivos y 3 verdaderos negativos. En la carga media 4 verdaderos positivos y 6 verdaderos negativos, y por último en la carga alta se obtuvo 8 verdaderos positivos y 12 verdaderos negativos. En ningún nivel se registró falsos positivos ni falsos negativos.

Resultados del consumo de CPU

El consumo de CPU registrado durante cada nivel de carga se presenta en la Tabla 9, con los valores promedio y máximos obtenidos en las condiciones evaluadas.

Tabla 9

Resultados del consumo de CPU según el nivel de carga

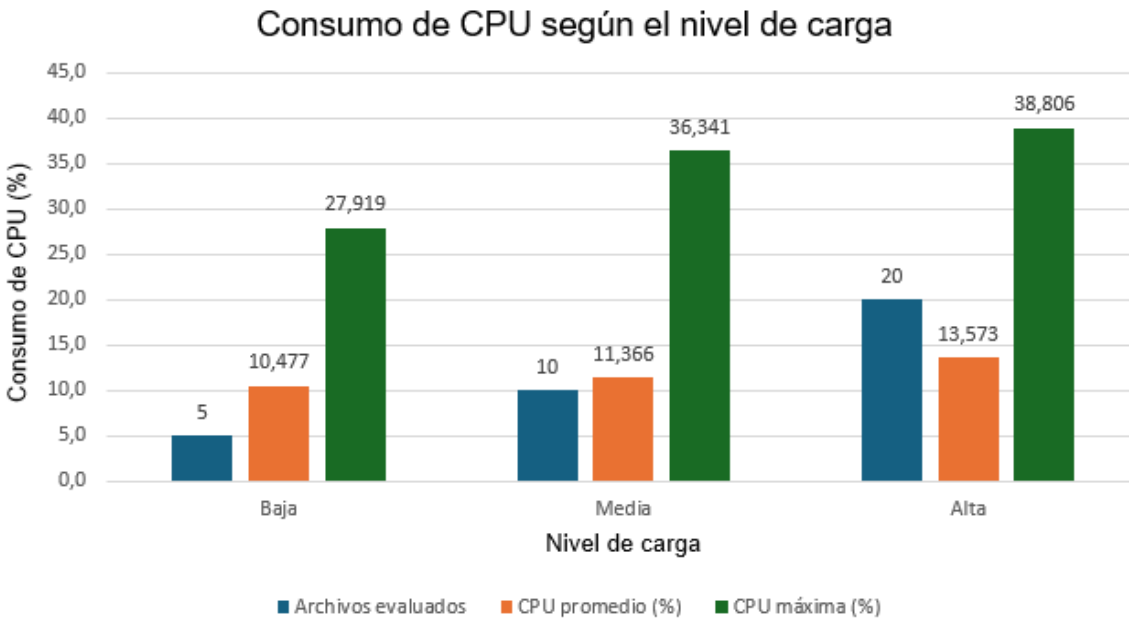
Nivel de carga	Archivos Evaluados	CPU Promedio (%)	CPU Máxima (%)
Baja	5	10.477	27,919
Media	10	11.366	36,341
Alta	20	13,573	38,806

Nota: Elaboración Propia

La variación del consumo promedio y máximo de CPU entre los niveles de carga se presenta en la Figura 18.

Figura 18

Comparación del consumo de CPU según el nivel de carga



Nota: Elaboración Propia

Los resultados indican el consumo de CPU, cuando se trataba de 5 archivos el consumo promedio de CPU fue de 10.477 %. Con una cantidad media de archivos el consumo promedio de CPU es de 11.366 %. Sin embargo, cuando se evalúan 20, el consumo promedio de CPU se incrementó un 13.573 %. Mientras que los valores máximos fueron de 27,919 % en la carga baja, 36,341 % en la carga media y 38,806 % en la carga alta.

Resultados del consumo de memoria

En la Tabla 10 se presenta el porcentaje de memoria durante la ejecución de cada nivel de carga, esto con el fin de evaluar el uso de recursos durante el procesamiento.

Tabla 10

Resultados del consumo de memoria según el nivel de carga.

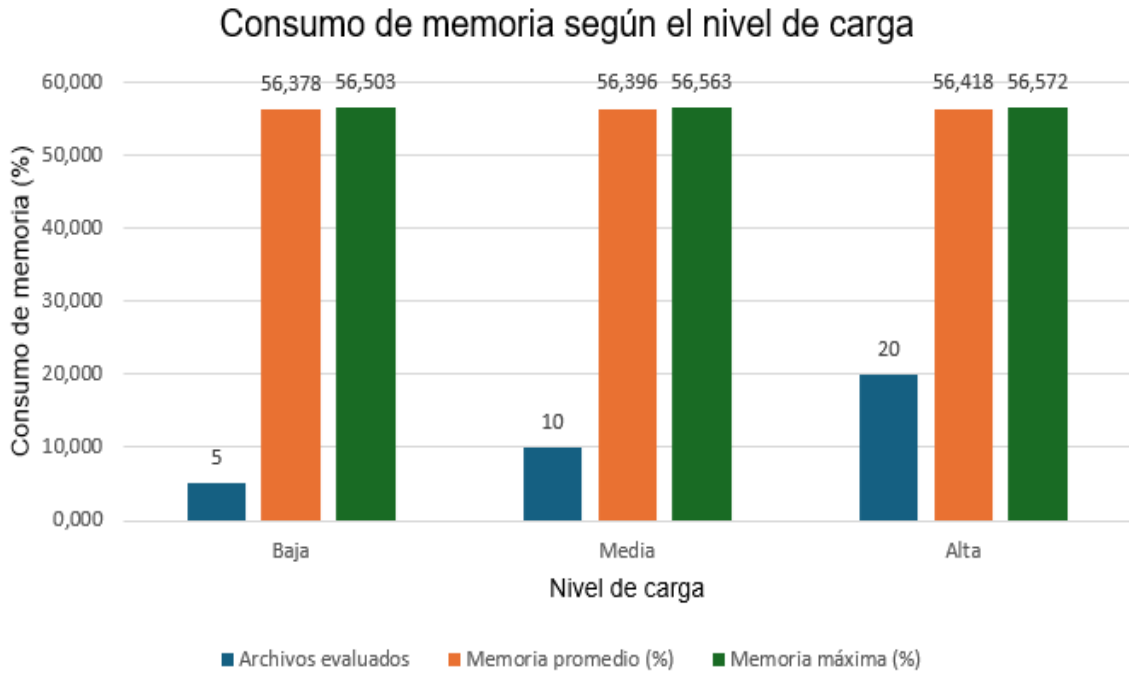
Nivel de carga	Archivos Evaluados	Memoria Promedio (%)	Memoria Máxima (%)
Baja	5	56,378	56,503
Media	10	56,396	56,563
Alta	20	56,418	56,572

Nota: Elaboración Propia.

El comportamiento del consumo promedio y máximo de memoria durante las pruebas se observa en la Figura 19.

Figura 19

Comparación del consumo de memoria según el nivel de carga



Nota: Elaboración Propia

Los resultados muestran la memoria utilizada en los tres niveles de carga. En la carga baja el promedio fue de 56,378 %. En la carga media fue de 56,396 %, y en la carga alta el promedio fue de 56,418 %. La diferencia entre la carga baja y la carga alta era de solo 0,040 puntos porcentuales.

Los valores máximos también cambiaron poco ya que en la carga baja el máximo fue del 56,503 %. En la carga media fue del 56,563 % y en la carga alta el máximo llegó al 56,572%.

Resultados del escenario de fuerza bruta SSH

Este escenario se implementó como validación funcional complementaria con ocho intentos fallidos de acceso desde una única dirección IP con un intervalo de 120 segundos. La Tabla 11 presenta los principales resultados obtenidos de esta prueba.

Tabla 11

Resultados del escenario de fuerza bruta SSH

Elemento evaluado	Resultado
Intentos fallidos detectados	8
Intervalo de observación	120 segundos
Dirección IP analizada	198.51.100.77
Coincidencia en MISP	Sí
Alerta generada	Sí

Caso registrado en TheHive	Sí
Dirección IP marcada como indicador	Sí
Tiempo de respuesta	15,844 segundos
Bloqueo automático	No aplicado
Estado final	Exitoso

Nota: Elaboración Propia.

Este escenario de fuerza bruta SSH se detectaron ocho intentos fallidos dentro de un tiempo de 120 segundos, vinculados a la dirección IP 198.51.100.77 y se localizó una coincidencia con la información registrada en MISP, se generó la alerta, se registró el caso en TheHive. y el registro de la dirección IP como indicador, todo esto en un tiempo de respuesta de 15.844 segundos.

Resultados del escenario de movimiento no autorizado de un archivo protegido.

Los resultados del escenario relacionado con el movimiento de un archivo protegido fuera de su ubicación autorizada se presentan en la Tabla 12, junto con las evidencias obtenidas durante la prueba.

Tabla 12

Resultados del escenario de movimiento no autorizado de un archivo protegido.

Elemento evaluado	Resultado
Archivo utilizado	PDF simulado de documentación interna
Ubicación autorizada	Carpeta de prueba definida para documentos protegidos
Evento simulado	Movimiento hacia una ubicación no permitida
Hashes analizados	SHA-256, SHA-1 y MD5
Coincidencias en MISP	No
Alerta generada	Sí
Caso registrado en TheHive	Sí
Criterio de respuesta	Incumplimiento de la política de ubicación
Acción aplicada	Traslado a cuarentena
Estado final	Exitoso

Nota: Elaboración Propia.

En el escenario de movimiento no autorizado de un archivo protegido, el documento de prueba fue detectado fuera de la ubicación definida como autorizada. A continuación, se revisaron sus valores SHA-256, SHA-1 y MD5 mediante Cortex y se efectuó la consulta correspondiente en MISP. Ninguno de los hashes presentó coincidencias con los indicadores registrados. Sin embargo, la ausencia de coincidencias no impidió la continuidad del flujo, debido a que la respuesta se basó en el incumplimiento de la política de ubicación establecida para el archivo. Como resultado, se generó la alerta, el caso fue registrado en TheHive y el

documento fue trasladado temporalmente a cuarentena como medida de contención y preservación de evidencia.

Resultados de las pruebas de persistencia y recuperación

Para asegurar de que todo el ambiente funciona después de una interrupción controlada, se reiniciaron los componentes principales. Luego se verificó que los servicios se vuelvan a levantar, que el agente se conecte y que el flujo automatizado se ejecute bien después de eso. Los resultados de las pruebas de persistencia y recuperación se presentan en la Tabla 13, con el estado de los servicios y del flujo después de los reinicios efectuados.

Tabla 13

Resultados de las pruebas de persistencia y recuperación

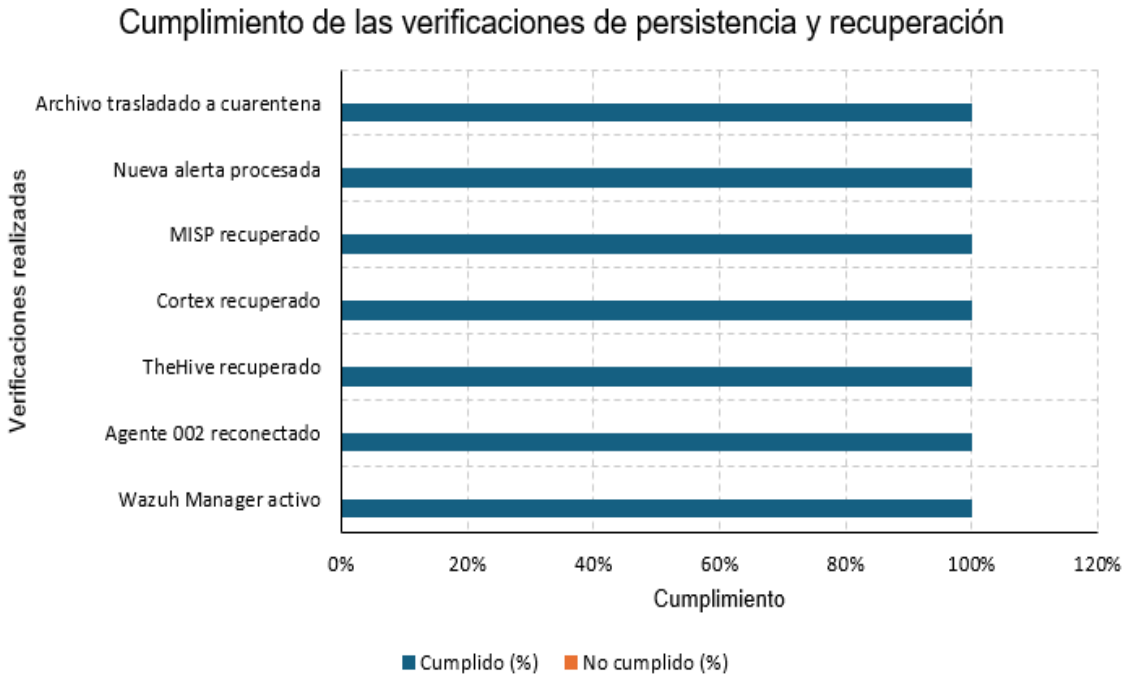
Prueba realizada	Elemento evaluado	Tiempo u observación
Reinicio del servicio central	Wazuh Manager	El servicio volvió al estado activo
Reconexión del agente	Agente 002	Recuperó la comunicación con el servidor
Reinicio de los servicios contenerizados	Entorno Docker	Se recuperaron 3 de 3 servicios en 56 segundos
Nueva prueba posterior al reinicio	Flujo automatizado	Se generó una nueva alerta y el archivo fue trasladado a cuarentena
Medición de la nueva ejecución	Tiempo de respuesta	21,82 segundos

Nota: Elaboración Propia

El cumplimiento de las verificaciones efectuadas después del reinicio controlado se representa en la Figura 20

Figura 20

Cumplimiento de las verificaciones realizadas después del reinicio.



Nota: Elaboración Propia

En la prueba de persistencia y recuperación, Wazuh Manager volvió al estado activo y el agente 002 recuperó la comunicación con el servidor. Los tres componentes principales desplegados en contenedores, TheHive, Cortex y MISP, se restablecieron en 56 segundos. Posteriormente, se generó una nueva prueba mediante la incorporación de otro archivo en la ruta monitoreada. El flujo procesó la nueva alerta, trasladó el archivo a cuarentena y registró un tiempo de respuesta de 21,82 segundos.

Discusión de los resultados

En este apartado se interpretan los resultados obtenidos durante la evaluación del entorno implementado. La discusión examina la variación de los tiempos de detección y

respuesta ante los niveles de carga, el grado de automatización, la clasificación de los eventos, el consumo de recursos y la recuperación posterior a los reinicios.

Comparación del tiempo medio de detección entre los distintos niveles de carga

La figura evidencia que el MTTD aumentó de manera continua al duplicarse la cantidad de archivos entre los niveles de carga. La separación más amplia entre las barras media y alta muestra que el ingreso simultáneo de 20 archivos tuvo una influencia más visible sobre el tiempo requerido para registrar las alertas.

Los 35 archivos generaron su alerta y esto es coherente con el papel de Wazuh como fuente de detección en la arquitectura y con el enfoque SOAR. Este busca gestionar un mayor volumen de eventos mediante procesos automatizados. Los resultados muestran que Wazuh funciona bien dentro del rango evaluado. Sin embargo, no se puede concluir de igual manera con cargas superiores a 20 archivos sin hacer pruebas adicionales.

Comparación del tiempo medio de respuesta entre los distintos niveles de carga

El MTTR aumentó más que el tiempo de detección. Esto significa que el crecimiento de la carga tuvo mayor impacto en las etapas posteriores a la alerta. En esas etapas se incluyen la revisión del indicador, la clasificación y la aplicación de la cuarentena. Después de que se ingresaron varios archivos que coincidían casi al mismo tiempo, las respuestas se dieron de forma gradual. Por eso, los últimos eventos tuvieron un tiempo de espera más largo.

Sin embargo, el tiempo final también depende de cómo se organizan las tareas y de cómo se comunican los componentes. En el entorno que se utiliza, el aumento del MTTR no provocó respuestas incompletas. Los 14 archivos que coincidieron terminaron de forma correcta en cuarentena. Entonces, la automatización fue efectiva en las cargas que se evaluaron. Pero los resultados muestran que la etapa de respuesta es más afectada cuando hay muchos eventos al mismo tiempo, en comparación con la etapa de detección inicial.

Niveles de Automatización alcanzado

El nivel de automatización se revisó durante el procesamiento de los 35 archivos, que estaban distribuidos en cargas baja, media y alta. Las ocho tareas definidas se realizan de forma automática desde la detección del evento hasta el registro de la evidencia. En los archivos que tuvieron coincidencias, el proceso continuó con el traslado a cuarentena. En los archivos que no tuvieron coincidencia, el proceso terminó tras el análisis y la clasificación sin realizar esa acción.

El resultado del 100 % muestra que, bajo las condiciones y cargas analizadas, no fue necesario mover información manualmente entre Wazuh, TheHive, Cortex y MISP ni iniciar las acciones planificadas. Este porcentaje se basa únicamente en las actividades probadas en el laboratorio.

Clasificación de los resultados

Los verdaderos positivos corresponden a los 14 archivos cuyos hashes estaban registrados en MISP y fueron identificados de manera correcta con coincidencia, lo que permitió ejecutar la acción de cuarentena. Los verdaderos negativos fueron los 21 archivos cuyos hashes no estaban registrados y que el entorno clasificó correctamente sin coincidencia, por lo que permanecieron en la ruta supervisada.

Los falsos positivos habrían ocurrido si un archivo sin coincidencia hubiera sido clasificado erróneamente como una amenaza conocida y trasladada a cuarentena. Por otra parte, los falsos negativos se habrían presentado si un archivo registrado en MISP no hubiera sido reconocido por el entorno. Durante las pruebas no se registró ninguno de estos dos errores es decir que los 35 archivos recibieron la clasificación esperada.

Comparación del consumo promedio de CPU según el nivel de carga

La figura muestra que el consumo promedio de CPU varió según la cantidad de archivos procesados, con un valor de 10,477 % en la carga baja a 13,573 % en la carga alta. De igual manera, el consumo máximo se elevó de 27.919 % a 38.806 %.

La diferencia entre los valores promedio y máximos refleja la presencia de picos temporales de procesamiento durante la ejecución de las cargas. Sin embargo, el valor más alto se mantuvo por debajo del 40 % y no se registraron interrupciones, por lo que no se evidenció una saturación de CPU en los niveles evaluados.

Comparación del consumo promedio de memoria según el nivel de carga

El consumo de memoria se considera estable en los tres niveles de carga del sistema. Aunque los archivos pasaron de 5 a 20 el promedio sólo varió de 56,378% a 56,418%, con una diferencia de 0,040 puntos porcentuales. Los valores máximos tuvieron un comportamiento similar, con una diferencia de tan sólo 0,069 puntos porcentuales entre la carga baja y la carga alta. Tampoco se registraron interrupciones durante las pruebas, por lo que no se evidenció una saturación de este recurso dentro de las cargas evaluadas.

Cumplimiento de las verificaciones de persistencia y recuperación

Tras el reinicio controlado, Wazuh Manager volvió a estar activo, el agente 002 recuperó la comunicación y TheHive, Cortex y MISP quedaron restablecidos en 56 segundos. Después se realizó otra prueba, se procesó la alerta y el archivo se trasladó a cuarentena en 21,82 segundos.

No sólo se comprobó la recuperación con la disponibilidad de los componentes, sino también mediante la ejecución completa de un nuevo flujo. Esto permitió comprobar que las configuraciones, la comunicación entre herramientas y la acción de respuesta seguían operativas tras el reinicio. Los dos tiempos corresponden a mediciones distintas: 56 segundos a la recuperación de los componentes y 21,82 segundos al tiempo de respuesta de la prueba posterior.

Capítulo V

Conclusiones

La revisión documental determinó que la selección de tecnologías para un entorno SOAR debe tener en cuenta la especialización funcional, la compatibilidad entre componentes, la interoperabilidad mediante mecanismos de integración y la posibilidad de mantener un despliegue controlado. Estos criterios sustentaron la elección de Wazuh, TheHive, Cortex, MISP, Docker y Docker Compose, debido a que sus capacidades cubrieron las necesidades de detección, gestión, análisis, inteligencia de amenazas y administración de servicios definidas para el laboratorio. Como resultado, se obtuvo una base técnica coherente para orientar el diseño y la posterior implementación del entorno.

El diseño de una arquitectura híbrida de despliegue conformada por servicios instalados directamente en Ubuntu Server y componentes ejecutados mediante contenedores Docker. Esta decisión respondió a los requisitos técnicos de las herramientas utilizadas. Wazuh se mantuvo instalado directamente en el servidor para supervisar los eventos y ejecutar las acciones de respuesta sobre el sistema operativo, mientras que TheHive, Cortex, MISP y sus dependencias se desplegaron mediante Docker Compose. Es así que el flujo iniciaba con la detección en Wazuh, continuaba con la gestión de la alerta en TheHive y el análisis de los observables mediante Cortex y MISP, para finalmente aplicar la respuesta correspondiente. Esta distribución facilitó la compatibilidad, el aislamiento y la administración conjunta de las plataformas.

La integración desarrollada permitió comprobar que la instalación independiente de herramientas no constituye por sí sola un entorno de orquestación. El aporte principal se obtuvo mediante el intercambio de información entre Wazuh, TheHive, Cortex y MISP, lo que permitió relacionar la detección de eventos, la creación de casos, el análisis de observables, la consulta de indicadores y la ejecución de respuestas dentro de un mismo flujo. Las actividades evaluadas se completaron de manera automática tanto en eventos con coincidencia como

eventos sin coincidencia, lo cual redujo la necesidad de transferir información manual entre plataformas.

Mediante la evaluación se comprobó la funcionalidad del entorno bajo los escenarios establecidos. El escenario principal clasifico los 35 archivos de forma correcta, con 14 verdaderos positivos y 21 verdaderos negativos, además las ocho actividades evaluadas alcanzaron un nivel de automatización del 100%. El incremento del MTTD y la mayor intensidad del MTTR ante cargas superiores identifico la etapa de respuesta como el principal punto de sensibilidad del flujo. Los escenarios de acceso mediante SSH y movimiento de un archivo protegido confirmaron que la plataforma responde a eventos de distinta naturaleza. El consumo de recursos se mantuvo dentro de la capacidad disponible y la recuperación posterior al reinicio confirmó la persistencia de las configuraciones y la continuidad operativa.

Recomendaciones

Se sugiere implementar mecanismos de procesamiento no simultáneo o colas de eventos, con el objetivo de prevenir que cada solicitud tenga que esperar que la otra finalice. Esto permitiría disminuir la acumulación de tiempo observada al incrementarse la cantidad de eventos procesados.

Se recomienda aumentar el número de ejecuciones efectuadas en cada escenario y usar muestras más amplias. Con el fin de obtener promedios relevantes, evaluar las variaciones de los resultados y realizar comparaciones más exactas entre los diferentes tipos de incidentes.

Se recomienda mantener respaldos periódicos de los archivos de configuración, scripts de integración y volúmenes persistentes antes de realizar actualizaciones en las herramientas. Esta medida facilitaría la recuperación del entorno ante errores de configuración o problemas de compatibilidad entre versiones.

Se aconseja expandir los mecanismos de seguimiento del flujo y análisis del mismo a través del envió de notificaciones al correo o Telegram en el momento que se genere, finalice o presente errores durante la aplicación de respuestas. También se podría añadir analizadores relacionados con reputación de archivos, direcciones IP y dominios, además de servicios antivirus con el fin obtener más información sobre los eventos y facilitar su revisión por parte del equipo analista.

Trabajos Futuros

Como una dirección futura del trabajo, se sugiere repartir los elementos de la arquitectura en servidores o máquinas virtuales independientes. Wazuh, TheHive, Cortex y MISP podrían establecerse de manera separada para analizar la comunicación entre las diferentes plataformas, la separación de los servicios, la disponibilidad del entorno y su capacidad de recuperación en caso de que uno de los componentes falle.

Como segunda línea de trabajo, se propone ampliar la muestra de monitoreo a un grupo más grande de equipos como el Departamento de Ciencias de la Computación. Esta ampliación permitiría evaluar cómo se comporta el entorno cuando varios dispositivos generan eventos. También permitiría revisar el aumento de alertas y ver si el sistema puede funcionar en un escenario institucional más grande.

Finalmente, se plantea incorporar dispositivos de red, como routers y switches, como nuevas fuentes de eventos. La recopilación de registros procedentes de estos equipos permitiría analizar intentos de acceso, cambios de configuración, conexiones sospechosas y posibles movimientos de información dentro de la red del laboratorio.

Referencias:

A. Nelson, S. R. (2025). *Incident Response Recommendations and Considerations for Cybersecurity.* Gaithersburg, MD.: NIST.

Amine, S. M. (23 de Julio de 2023). *TheHive 5 Incident Management System: Enhancing Cybersecurity Resilience and Collaboration*. Obtenido de Medium: https://mohamedsaidani34.medium.com/thehive-5-incident-management-system-enhancing-cybersecurity-resilience-and-collaboration-b8fe651d3407

Banco Santander. (2020). *¿Qué es un incidente de seguridad?* Obtenido de Banco Santander: https://www.bancosantander.es/glosario/incidente-seguridad

Borges, E. (2021). *What is the MISP Threat Sharing Platform?* Obtenido de Recordedfuture.com: https://www.recordedfuture.com/threat-intelligence-101/tools-and-technologies/misp

Bridges, R. A., Rice, A. E., Oesch, S., Nichols, J. A., Watson, C., Spakes, K., . . . Erwin, S. (12 de Agosto de 2022). *Testing SOAR Tools in Use.* Obtenido de https://arxiv.org/abs/2208.06075

Centro de Gestión de Incidentes Informáticos. (2025). *Malware Information Sharing Platform*. Obtenido de Centro de Gestión de Incidentes Informáticos: https://www.csirt.gob.bo/es/misp

Chouliaras, N., Kantzavelou, I., Maglaras, L., Pantziou, G., & Ferrag, M. A. (2023). A novel autonomous container-based platform for cybersecurity training and research. *PeerJ Computer Science*, e1574.

Cisco. (2017). *What is cybersecurity?* Obtenido de Cisco: https://www.cisco.com/site/us/en/learn/topics/security/what-is-cybersecurity.html

Docker. (2018). *Docker Compose*. Obtenido de Docker Docs: https://docs.docker.com/compose/

Docker. (2018). *What is a Container?* Obtenido de Docker: https://www.docker.com/resources/what-container/

Fortinet. (2022). *Indicadores de compromiso*. Obtenido de Fortinet: https://www.fortinet.com/lat/resources/cyberglossary/indicators-of-compromise

Groenewegen, A. &. (Junio de 2021). *University of Amsterdam.* Obtenido de TheHive Project: The maturity of an open-source Security Incident Response platform: https://www.researchgate.net/publication/352715439_TheHive_Project_The_maturity_of_an_open-source_Security_Incident_Response_platform

Guduru, S. (2022). SOAR Automation: Palo Alto Cortex XSOAR Playbooks for MISP Threat Intel Enrichment and TheHive Project Integration. *International Journal of Science and Research (IJSR)*, 4.

IBM. (2019). *What is SIEM?* Obtenido de IBM Think: https://www.ibm.com/think/topics/siem

IBM. (2025). *¿Qué es el tiempo medio de reparación (MTTR)?* Obtenido de IBM Think: https://www.ibm.com/es-es/think/topics/mttr

Irvine, C. E., Thompson, M. F., McCarrin, M., & Khosalim, J. (2017). Live Lesson: Labtainers: A Docker-based Framework for Cybersecurity Labs. *2017 USENIX Workshop on Advances in Security Education (ASE 17)* (págs. 1-8). Vancouver, BC: USENIX Association.

Kafami, M. (4 de Abril de 2025). *What is XSOAR and How is it Used?* Obtenido de CBT Nuggets: https://www.cbtnuggets.com/blog/technology/security/what-is-xsoar-and-how-is-it-used

Kalonji, K. (2022). *Capabilities and Use of Cortex as Part of the DYNAMO Project.* Vantaa, Finlandia: Laurea University of Applied Sciences.

Kaspersky. (2024). *¿Qué es la inteligencia de amenazas? Definición y explicación*. Obtenido de Kaspersky: https://latam.kaspersky.com/resource-center/definitions/threat-intelligence

Merkel, D. (26 de Diciembre de 2018). Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal*, 5.

Microsoft. (26 de 2025). *What are unified security operations in the Microsoft Defender portal*. Obtenido de Microsoft Learn: https://learn.microsoft.com/en-us/unified-secops/overview-unified-security

MISP. (2022). *MISP Open Source Threat Intelligence Platform & Open Standards For Threat Intelligence Sharing*. Obtenido de MISP Intelligence: https://www.misp-project.org/

Muriyatmoko, D., Tjahyanto, A., Harmini, T., & Iqbal, M. R. (2024). Muriyatmoko, D.; Tjahyanto, A.; Harmini, T.; Iqbal, M. R. *Procedia Computer Science*, 234.

Nelson, A., Rekhi, S., Souppaya, M., & Scarfone, K. (2025). *Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile.* https://doi.org/10.6028/NIST.SP.800-61r3: National Institute of Standards and Technology.

Norem, S., Rice, A. E., Erwin, S., Bridges, R. A., Oesch, S., & Weber, B. (1 de diciembre de 2021). *A Mathematical Framework for Evaluation of SOAR Tools with Limited Survey Data.* Obtenido de https://arxiv.org/abs/2112.00100

Oracle. (24 de Febrero de 2025). *¿Qué es una API (Application Programming Interface)?* Obtenido de Oracle: https://www.oracle.com/lad/cloud/cloud-native/api-management/what-is-api/

Palo Alto Networks. (2024). *¿Qué es SOAR?* Obtenido de Cyberpedia: https://www.paloaltonetworks.lat/cyberpedia/what-is-soar

Pereira, V. R. (2023). *Cyber-attack Detection and Response Using Open-Source Tools.* Irlanda: National College of Ireland.

Pulyala, S. R., Desetty, A. G., & Jangampet, V. D. (2019). The Impact of Security Orchestration, Automation, and Response (SOAR) on Security Operations Center (SOC) Efficiency: A

Comprehensive Analysis. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 1545-1549.

SentinelOne. (3 de Abril de 2014). *What Is MTTD (Mean Time to Detect)? A Detailed Explanation*. Obtenido de SentinelOne: https://www.sentinelone.com/blog/mttd-mean-time-to-detect-detailed-explanation/

Srinithi. (27 de febrero de 2026). *¿Qué son las herramientas de SOAR y por qué son importantes en 2026?* Obtenido de ManageEngine: https://www.manageengine.com/latam/log-management/soar/que-son-las-herramientas-de-soar-y-por-que-son-importantes-en-2026.html

Sworna, Z. T., Islam, C., & Babar, M. A. (20 de enero de 2022). *APIRO: A Framework for Automated Security Tools API Recommendation.* Obtenido de https://arxiv.org/abs/2201.07959

TecnoSolucion. (2022). *¿Qué son los Libros de Estrategias (Playbooks) y cómo pueden ayudar al manejo de proyectos a través de la mensajería instantánea empresarial?* Obtenido de TecnoSoluciones.com: https://tecnosoluciones.com/que-son-los-libros-de-estrategias-playbooks-y-como-pueden-ayudar-al-manejo-de-proyectos-a-traves-de-la-mensajeria-instantanea-empresarial/

Wagner, C. D. (2016). MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform. *ACM DIGITAL LIBRARY*, 8.

Wagner, C., Dulaunoy, A., Wagener, G., & Iklody, A. (2016). MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform. *Tercer Taller de la ACM sobre Intercambio de Información y Seguridad Colaborativa (WISCS 2016)*, (pág. 6). Vienna, Austria.

Wazuh. (2018). *Components*. Obtenido de Wazuh: https://wazuh.com/platform/