**UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ**

**Colegio de Ciencias e Ingenierías**

**Orquestador de Agentes de IA para gestión de incidentes de ciberseguridad**

.

**José Luis Contreras Parreño**

**Ing. Ciencias de la Computación**

Trabajo de fin de carrera presentado como requisito

para la obtención del título de

Ingeniero en Ciencias de la Computación

Quito, de mes de 2025

**Universidad San Francisco de Quito USFQ**

**Colegio de Ciencias e Ingenierías**

HOJA DE CALIFICACIÓN

DE TRABAJO DE FIN DE CARRERA

**Orquestador de Agentes de IA para gestión de incidentes de ciberseguridad**

José Luis Contreras Parreño

**Nombre del profesor, Título académico Roberto Andrade,** PhD. in Informatics

Quito, día de mes de 2025

**© DERECHOS DE AUTOR**

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en la Ley Orgánica de Educación Superior del Ecuador.

Nombres y apellidos: José Luis Contreras Parreño

Código: 00203046

Cédula de identidad: 1718683863

Lugar y fecha: Quito, día de mes de 2025

**ACLARACIÓN PARA PUBLICACIÓN**

**Nota:** El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en <http://bit.ly/COPETheses>.

**UNPUBLISHED DOCUMENT**

**Note:** The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on <http://bit.ly/COPETheses>.

RESUMEN

Este trabajo presenta el diseño e implementación de un orquestador de agentes de inteligencia artificial (IA) para la gestión de incidentes de ciberseguridad. La investigación parte de las limitaciones de los sistemas tradicionales SIEM y SOAR, los cuales requieren alta intervención humana, integraciones costosas y presentan tiempos elevados de respuesta. El proyecto propone un enfoque multi-agente que combina Modelos de Lenguaje de Gran Escala (LLMs) y técnicas de Recuperación Aumentada por Generación (RAG). La arquitectura diseñada incluye agentes especializados en monitoreo, análisis y respuesta, coordinados mediante n8n y LanGraph si se considera pertinente. Los resultados esperados incluyen una reducción del MTTR de al menos un 20% y una disminución de falsos positivos en un 15% respecto a la línea base. Este trabajo busca demostrar la viabilidad de soluciones flexibles, escalables y de menor costo operativo para la gestión de incidentes en entornos de seguridad corporativa y de servicios.

Palabras clave: ciberseguridad, inteligencia artificial, orquestación multi-agente, SIEM, SOAR, LLM, RAG, LanGraph, n8n.

ABSTRACT

This work presents the design and implementation of an AI agent orchestrator for cybersecurity incident management. The research starts from the limitations of traditional SIEM and SOAR systems, which require high human intervention, costly integrations, and elevated response times. The project proposes a multi-agent approach combining Large Language Models (LLMs), Retrieval-Augmented Generation (RAG). The designed architecture includes monitoring, analysis, and response agents, coordinated through LangGraph and n8n workflows. Expected results include reducing MTTR by at least 20% and decreasing false positives by 15% compared to the baseline. This project aims to demonstrate the feasibility of flexible, scalable, and cost-effective solutions for incident management in enterprise and service environments.

Key words: cybersecurity, artificial intelligence, multi-agent orchestration, SIEM, SOAR, LLM, RAG, LangGraph, n8n.

TABLA DE CONTENIDO

Introducción 9

Estado del arte 11

Comparación de Esquemas de Eventos 12

Limitaciones y Capacidades de los Sistemas SIEM y SOAR Tradicionales 13

Capacidades Fundamentales de SIEM y SOAR 13

Limitaciones en Correlación de Eventos y Reducción de Falsos Positivos 13

Principios y Arquitecturas de Orquestación Multi-Agente Basadas en LLMs y RAG 14

Fundamentos de RAG y Orquestación Multi-Agente 14

Patrones Arquitectónicos para Respuesta en Tiempo Real 14

Potencial de la Orquestación Multi-Agente Basada en LLMs y RAG para la Gestión de Incidentes de Ciberseguridad: Un Análisis Comparativo con SIEM y SOAR Tradicionales 16

DDoS Attack Incident Response Playbook 23

Step 1: Identification 23

Step 2: Containment 23

Step 3: Eradication 23

Step 4: Recovery 23

Step 5: Lessons Learned 24

Diferencias más importantes 29

Descripción del flujo por nodo. 31

Conclusiones 45

Referencias bibliográficas 46

Anexo A: Título 47

Anexo B: Título 47

Anexo C: Título 48

ÍNDICE DE TABLAS

(Este índice se incluye únicamente si el estudiante utiliza tablas en el contenido)

En el cuerpo del documento, los títulos de las tablas deben utilizar el formato:

Tabla #. Título de la tabla (debe ser autodescriptivo y no debe depender del texto) …..12

ÍNDICE DE FIGURAS

(Este índice se incluye únicamente si el estudiante utiliza figuras en el contenido)

En el cuerpo del documento, los títulos de las tablas deben utilizar el formato:

Figura #. Título de la figura (debe ser autodescriptivo y no debe depender del texto)….. 12

# Introducción

La gestión de incidentes de ciberseguridad se ha convertido en un desafío crítico en un contexto donde la complejidad de las infraestructuras tecnológicas y la sofisticación de los ataques avanzan a un ritmo sin precedentes. Informes recientes de la Cybersecurity and Infrastructure Security Agency (CISA, 2024) y de la European Union Agency for Cybersecurity (ENISA, 2023) destacan que el volumen de incidentes de seguridad gestionados por centros de operaciones de seguridad (SOC) ha incrementado en alrededor del 30% anual. Este crecimiento genera una sobrecarga para analistas humanos, quienes deben filtrar, clasificar y priorizar cientos o miles de alertas diarias.

Los sistemas tradicionales SIEM (Security Information and Event Management) centralizan registros y correlacionan eventos de distintas fuentes, brindando una primera capa de visibilidad. Sin embargo, sus limitaciones son claras: requieren intervención humana constante, sus correlaciones se basan en reglas estáticas, y presentan dificultades para adaptarse a amenazas emergentes (Islam, Babar & Nepal, 2019). Por su parte, las plataformas SOAR (Security Orchestration, Automation and Response) incorporan flujos de automatización con “playbooks” predefinidos, mejorando los tiempos de reacción, pero su efectividad depende de integraciones costosas y rígidas (Kremer et al., 2023).

En este contexto surge la propuesta de diseñar un orquestador de agentes de inteligencia artificial (IA), donde múltiples agentes especializados en monitoreo, análisis y respuesta colaboran entre sí bajo mecanismos de coordinación estructurados. El presente trabajo busca demostrar que un enfoque multi-agente puede:

* Reducir el MTTR (Mean Time to Resolution) en al menos un 20%.
* Disminuir falsos positivos en un 20% frente a una línea base.
* Ofrecer explicaciones trazables para cada decisión de análisis y respuesta.

La relevancia del proyecto es doble: por un lado, podría significar una ayuda en respuesta a la necesidad de soluciones escalables y adaptables frente a la evolución constante de ciberataques; por otro, plantea una alternativa viable para contextos corporativos y de servicios en países en desarrollo, donde el acceso a licencias propietarias resulta limitado debido al alto costo.

# Estado del arte

La gestión de incidentes de ciberseguridad ha evolucionado de manera significativa en las últimas dos décadas. El crecimiento exponencial de los ataques dirigidos y la complejidad de las infraestructuras tecnológicas obligaron a pasar de enfoques manuales a plataformas centralizadas y posteriormente a sistemas automatizados.

La creciente sofisticación y volumen de los incidentes de ciberseguridad ha puesto en evidencia las limitaciones de los sistemas tradicionales de gestión de eventos e información de seguridad (SIEM) y de orquestación, automatización y respuesta de seguridad (SOAR). Estos sistemas, aunque fundamentales, enfrentan retos significativos en la correlación de eventos, reducción de falsos positivos y adaptación a amenazas emergentes. La necesidad de enfoques más adaptativos, escalables y automatizados es cada vez más indispensable.

Mientras que SIEM y SOAR han sido la columna vertebral de los centros de operaciones de seguridad (SOC), la irrupción de modelos de lenguaje de gran escala (LLMs) y arquitecturas de generación aumentada por recuperación (RAG) ha abierto nuevas posibilidades. La orquestación multi-agente basada en LLMs y RAG promete superar las limitaciones de los sistemas tradicionales, integrando razonamiento contextual, aprendizaje continuo y respuesta autónoma.

Un Centro de Operaciones de Seguridad (SOC) es una unidad centralizada responsable de monitorear, analizar y defender la postura de seguridad de una organización. Los SOC desempeñan un papel fundamental en la protección de los activos digitales, como los datos, las aplicaciones y la infraestructura, frente a las ciberamenazas. La cual es realizada a través de una combinación de personas, procesos y tecnologías, incluyendo herramientas sofisticadas como sistemas de gestión de eventos e información de seguridad (SIEM), firewalls, sistemas de detección de intrusos y algoritmos de aprendizaje automático.

**Esquema Canónico de Eventos (CEC) y Comparación con Otros Esquemas**

El Esquema Canónico de Eventos (CEC) es una representación estructurada de eventos que permite modelar el conocimiento del mundo sobre la progresión típica de eventos. Este esquema se puede inducir utilizando modelos de lenguaje preentrenados (LLMs), lo que simplifica el proceso de inducción y permite manejar relaciones jerárquicas y temporales de manera eficiente. A continuación, se presenta una comparación del CEC con otros esquemas de eventos como CEF de ArcSight, Syslog y STIX.

### Comparación de Esquemas de Eventos

|  |  |  |  |
| --- | --- | --- | --- |
| **Esquema** | **Descripción** | **Ventajas** | **Desventajas** |
| **CEC** | Utiliza LLMs para inducir esquemas de eventos a partir de texto no etiquetado. | - Simplificación del proceso de inducción. <br> - Manejo eficiente de relaciones jerárquicas y temporales. <br> - Alta calidad y cobertura de esquemas. | - Dependencia de modelos de lenguaje preentrenados. |
| **CEF (ArcSight)** | Formato de evento común utilizado para la normalización de eventos de seguridad. | - Estandarización en la industria de seguridad. <br> - Facilita la integración con múltiples sistemas de seguridad. | - Limitado a eventos de seguridad. <br> - Menos flexible para otros tipos de eventos. |
| **Syslog** | Protocolo estándar para el envío de mensajes de registro en una red IP. | - Amplia adopción y soporte en sistemas operativos y dispositivos de red. <br> - Simplicidad y eficiencia en la transmisión de mensajes. | - Formato de mensaje limitado. <br> - Falta de estructura para eventos complejos. |
| **STIX** | Estandarización para el intercambio de información sobre amenazas cibernéticas. | - Estructura detallada y rica en contexto. <br> - Facilita el intercambio de información entre organizaciones. | - Complejidad en la implementación. <br> - Enfoque específico en ciberseguridad. |

Tabla 1. Comparación CEC

* **Calidad y Cobertura**: El CEC, al utilizar LLMs, puede inducir esquemas de alta calidad y cobertura en diversos dominios, superando las limitaciones de esquemas específicos como CEF y STIX que están más enfocados en la seguridad.
* **Flexibilidad**: A diferencia de Syslog y CEF, que tienen formatos más rígidos y específicos, el CEC ofrece una mayor flexibilidad para representar eventos complejos y sus relaciones.
* **Eficiencia**: El uso de LLMs en el CEC permite una inducción más eficiente y precisa de esquemas de eventos, mejorando la calidad de las relaciones temporales y jerárquicas en comparación con métodos tradicionales.

A continuación, se lista características respecto a los sistemas SIEM y SOAR, sus capacidades y limitaciones:

### Limitaciones y Capacidades de los Sistemas SIEM y SOAR Tradicionales

### Capacidades Fundamentales de SIEM y SOAR

* **SIEM:** Centraliza la recolección, correlación y análisis de eventos de seguridad, permitiendo la detección y priorización de amenazas. Sin embargo, depende de reglas y firmas, lo que limita su capacidad ante amenazas desconocidas.
* **SOAR:** Automatiza flujos de respuesta, mejorando la velocidad y consistencia, pero requiere personalización y supervisión humana significativa.

### Limitaciones en Correlación de Eventos y Reducción de Falsos Positivos

* **Desafíos en integración de datos heterogéneos:** Dificultad para normalizar y correlacionar eventos de múltiples fuentes, especialmente en grandes despliegues.
* **Fatiga de alertas:** Alto volumen de falsos positivos, lo que sobrecarga a los analistas y reduce la eficacia de la respuesta.
* **Limitaciones de reglas:** Incapacidad para adaptarse rápidamente a nuevas tácticas de ataque.

### Principios y Arquitecturas de Orquestación Multi-Agente Basadas en LLMs y RAG

### Fundamentos de RAG y Orquestación Multi-Agente

* **Agentic RAG:** Extiende RAG tradicional permitiendo ingestión dinámica de datos y razonamiento en tiempo real mediante agentes especializados (Function Calling, ReAct, LLMCompiler).
* **Descentralización:** Arquitecturas RAG descentralizadas mejoran eficiencia y privacidad, permitiendo que entidades distribuidas gestionen recuperación, augmentación y generación.
* **Mitigación de alucinaciones:** Integración de fuentes externas y protocolos de control para reducir errores y mejorar la fiabilidad.

### Patrones Arquitectónicos para Respuesta en Tiempo Real

* **Modularidad y escalabilidad:** Frameworks como NetSecGame permiten escenarios multi-agente complejos y toma de decisiones secuencial sin necesidad de reentrenamiento.
* **Integración con herramientas forenses:** Orquestación de agentes LLM con herramientas como PCAP readers y sistemas de recuperación de información para análisis y atribución de incidentes.
* **Seguridad y benchmarking:** Frameworks como Agent Security Bench (ASB) formalizan ataques y defensas, evaluando vulnerabilidades en agentes LLM.

Avances hacia arquitecturas multi-agente

Frente a las limitaciones de SIEM y SOAR, la investigación en ciberseguridad se ha orientado hacia arquitecturas basadas en agentes de IA, capaces de colaborar en tareas de detección, análisis y respuesta. Estos agentes, apoyados en Modelos de Lenguaje de Gran Escala (LLMs) y Recuperación Aumentada por Generación (RAG), han mostrado mejoras notables en precisión y eficiencia.

Un ejemplo destacado es CyberRAG, un sistema que integra RAG y LLMs para la clasificación de ataques. De acuerdo con Blefari et al. (2025), esta arquitectura logra más del 94% de precisión en entornos simulados y reduce en un 45% los tiempos de triaje en centros de operaciones de seguridad.

De manera similar, Audit-LLM introduce un mecanismo de debate entre agentes especializados para detectar amenazas internas en registros corporativos. Los resultados reportados por Song et al. (2024) muestran una reducción significativa de falsos positivos, alcanzando un rango de 3,7–6,7%, muy inferior al de seis sistemas de referencia evaluados.

Otros proyectos han explorado modelos de coordinación más complejos. El caso de Triangle, desarrollado por Tsinghua University y Microsoft, implementa agentes negociadores para la priorización de incidentes en entornos de nube. Este sistema demostró una reducción sustancial en el tiempo de toma de decisiones frente a métodos manuales (Tsinghua University et al., 2025). En paralelo, CyGATE emplea teoría de juegos para optimizar la planificación de parches y defensa adaptativa, integrando agentes basados en LLMs y RAG para anticipar los movimientos de un adversario (Jiang et al., 2025).

Coordinación y orquestación de agentes

Más allá del desempeño individual de cada agente, la literatura reciente enfatiza la importancia de los mecanismos de orquestación y coordinación. Herramientas como LangGraph permiten integrar múltiples agentes en flujos colaborativos, mientras que enfoques de ChatOps facilitan la interacción en tiempo real entre humanos y sistemas automatizados. En la misma línea, CyberGuardian 2 (Paduraru, Patilea & Stefanescu, 2025) evaluó la integración de agentes LLM en redes distribuidas. Sus resultados resaltan la modularidad como una característica esencial para la escalabilidad y resiliencia, ya que permiten añadir o reemplazar agentes sin comprometer el sistema global.

Un enfoque de orquestación multi-agente basado en LLMs y RAG puede mejorar significativamente la detección, priorización y respuesta a incidentes de ciberseguridad en comparación con los sistemas tradicionales SIEM y SOAR. Esto se debe a la capacidad de estos sistemas para integrar información contextual, reducir falsos positivos mediante técnicas avanzadas de aprendizaje automático y facilitar una respuesta más dinámica y adaptable ante amenazas emergentes. Sin embargo, es importante abordar los retos asociados, como la gestión de latencia, la seguridad frente a vulnerabilidades en la inyección de prompts y la integración con herramientas de seguridad preexistentes.

### Potencial de la Orquestación Multi-Agente Basada en LLMs y RAG para la Gestión de Incidentes de Ciberseguridad: Un Análisis Comparativo con SIEM y SOAR Tradicionales

|  |  |  |
| --- | --- | --- |
| **Aspecto a evaluar** | **SIEM/SOAR Tradicional** | **Multi-Agente LLM + RAG** |
| **Detección de Incidentes** | Basada en reglas y firmas; limitada ante amenazas nuevas; alta tasa de falsos positivos | Razonamiento contextual, integración dinámica de datos, reducción de falsos positivos mediante aprendizaje automático |
| **Priorización** | Priorización rígida, dependiente de reglas; limitada contextualización | Priorización adaptativa, análisis semántico y contextual, integración de fuentes heterogéneas |
| **Respuesta** | Automatización básica, requiere supervisión humana significativa | Respuesta dinámica, orquestación autónoma, integración con herramientas forenses y de seguridad |
| **Reducción de Falsos Positivos** | Mejoras con ML, pero limitada por reglas y datos heterogéneos | Aprendizaje profundo, clustering, razonamiento multi-agente, reducción significativa de alertas irrelevantes |
| **Escalabilidad y Adaptabilidad** | Escalabilidad limitada, integración compleja, adaptación lenta | Modularidad, escalabilidad nativa, adaptación en tiempo real a nuevas amenazas |
| **Privacidad y Seguridad** | Centralización, riesgos de brechas, limitaciones en privacidad | Gobernanza de datos, privacidad reforzada, integración de blockchain y MFA |
| **Latencia y Coste** | Latencia variable, costes altos en grandes despliegues | Latencia optimizable, costes ajustables según arquitectura y caching |
| **Transparencia y Confianza** | Limitada, dependiente de reporting manual | Evaluación centrada en usuario, métricas de transparencia y confianza |

Tabla 2. Análisis Comparativo SIEM VS SOAR

Sin embargo, las soluciones tradicionales aún presentan limitaciones frente al volumen, velocidad y sofisticación de las amenazas actuales, lo que ha motivado el desarrollo de arquitecturas basadas en agentes de inteligencia artificial (IA).

|  |  |  |  |
| --- | --- | --- | --- |
| Aspecto | Tradicional | Ansible | N8N |
| Tecnología Base | Scripts manuales, shell/bash, cron jobs, SIEM/SOAR con reglas estáticas. | Motor de automatización declarativo basado en YAML y Python; usa SSH, APIs REST y módulos predefinidos. | Motor de automatización visual basado en flujos de trabajo (Node.js, TypeScript), con nodos conectables por API. |
| Ventajas | Control manual total. - No depende de software externo. - Fácil de implementar en entornos pequeños. | Escalable y repetible. - Gran soporte para infraestructura como código (IaC). - Permite automatizar tareas de seguridad (patching, response, configuración). | Interfaz visual intuitiva. - Integración rápida con APIs, bases de datos y herramientas SOC (como Wazuh o Slack). - Bajo costo y alto grado de personalización. - Ideal para orquestar agentes LLM o flujos RAG. |
| Limitaciones | Requiere intervención humana constante. - Difícil de escalar y auditar. - Alta tasa de falsos positivos. | Requiere conocimiento técnico avanzado. - Falta de interfaz visual. - Limitado en tareas cognitivas (no procesa texto ni razonamiento contextual). | Depende de conectividad y APIs estables. - Menor control granular sobre sistemas locales. - No es nativo para automatizaciones de bajo nivel (como configuración de firewalls). |

Tabla 3. Ventajas, Limitaciones Tradicional vs Ansible vs N8N

Resultados y métricas de evaluación

Los estudios revisados coinciden en que las arquitecturas multi-agente ofrecen mejoras consistentes frente a los sistemas tradicionales:

* Precisión de detección superior al 94% (Blefari et al., 2025; Alshamrani, 2025).
* Reducción de falsos positivos al rango de 3,7–6,7% (Song et al., 2024).
* Correlaciones exitosas en la priorización de incidentes (MCC=0,998) (Roelofs et al., 2024).
* Reducción del MTTR entre un 35% y 45% (Lin et al., 2025).

Estas métricas evidencian una clara ventaja frente a SIEM y SOAR, que suelen depender de análisis humanos y procesos rígidos.

Limitaciones y desafíos

A pesar de los resultados prometedores, la literatura también reconoce importantes limitaciones. En primer lugar, gran parte de los estudios se desarrolla en entornos simulados, lo que limita la validez externa y dificulta la extrapolación a entornos empresariales complejos (Nyberg & Johnson, 2024). En segundo lugar, la constante evolución de los LLMs plantea un reto: modelos que hoy ofrecen resultados sobresalientes pueden volverse obsoletos en cuestión de meses, obligando a rediseñar marcos de integración constantemente.

Otro desafío identificado es la necesidad de estudios longitudinales y despliegues reales que evalúen la escalabilidad, resiliencia y costo total de propiedad de estos sistemas en ambientes de producción. Autores como Paduraru et al. (2025) y Roelofs et al. (2024) destacan que, aunque existen pruebas piloto exitosas, aún falta evidencia a gran escala para consolidar su adopción en sectores críticos.

**Desafíos:**  
 • **Fatiga de alertas:** Los analistas de los SOC suelen sufrir agotamiento debido al alto volumen de alertas, muchas de las cuales son falsos positivos.  
 • **Integración y personalización:** Existe dificultad para integrar diversas herramientas de seguridad y adaptarlas a las necesidades específicas de cada organización (9 10).  
 • **Uso de recursos:** Las herramientas de monitoreo tradicionales pueden consumir muchos recursos y, en algunos casos, no detectar vulnerabilidades de tipo *zero-day* (7).

**Soluciones:**  
 • **Automatización:** Implementar mecanismos automáticos y modelos de aprendizaje automático para reducir los procesos manuales y mejorar la detección de amenazas.  
 • **Arquitecturas avanzadas:** Utilizar marcos como λ-NF3 y NF3 para un análisis de datos y detección de anomalías más eficientes.  
 • **Colaboración humano-IA:** Mejorar las operaciones del SOC mediante una colaboración efectiva entre los analistas humanos y las herramientas impulsadas por inteligencia artificial.

**Tendencias emergentes**

• **Inteligencia Artificial y Aprendizaje Automático:** Creciente uso de la IA y el aprendizaje automático para automatizar y mejorar las funciones del SOC, incluyendo la monitorización en tiempo real y el triaje de alertas.  
 • **Plataformas SOAR:** Adopción de plataformas SOAR para optimizar las operaciones de seguridad, mejorar la respuesta ante incidentes y reducir la complejidad operativa.  
 • **Integración del IoT:** Los SOC se están adaptando al creciente número de dispositivos IoT, lo que requiere un inventario de activos en tiempo real y la interpretación de protocolos.

**Respuesta a incidentes (IR)**

La respuesta a incidentes (IR) es una función crítica dentro de la ciberseguridad, orientada a detectar, mitigar y prevenir amenazas dentro de una red. Implica un enfoque integral que abarca aspectos técnicos, de gestión, legales y sociales.Tradicionalmente, la respuesta a incidentes ha sido reactiva, centrándose en abordar los problemas técnicos a medida que ocurren (1 4). Sin embargo, las metodologías recientes enfatizan un enfoque más proactivo e integrado.

**Componentes clave de la respuesta a incidentes:**

1. **Detección e identificación:** El primer paso consiste en identificar y detectar los incidentes de seguridad. Esto requiere sistemas de monitoreo robustos y la capacidad de reconocer anomalías que puedan indicar una brecha.
2. **Contención y eliminación:** Una vez detectado el incidente, el siguiente paso es contener y eliminar la amenaza para evitar daños adicionales. Esto implica aislar los sistemas afectados y eliminar los elementos maliciosos.
3. **Recuperación y restauración:** Tras la contención, los esfuerzos se centran en recuperar y restaurar los sistemas comprometidos a su estado normal. Esto incluye la recuperación de datos y la reparación de sistemas.
4. **Investigación forense:** Para responsabilizar a los intrusos y comprender la brecha, se lleva a cabo una investigación forense. Esto implica recopilar y documentar evidencia durante todo el proceso de respuesta.
5. **Revisión y aprendizaje posterior al incidente:** Un aspecto crítico, a menudo pasado por alto, es la revisión posterior al incidente. Esta etapa implica analizar el evento para aprender de él y mejorar las respuestas futuras. El modelo de aprendizaje de doble bucle enfatiza la necesidad de abordar problemas sistémicos y fortalecer los procesos estratégicos de seguridad.

**Tendencias emergentes y desafíos:**

• **Aprendizaje proactivo y perspectivas socio-técnicas:** El método *Incident Response Management* (IRMA) integra la respuesta tradicional a incidentes con el aprendizaje proactivo y perspectivas socio-técnicas, con el objetivo de mejorar la postura general de seguridad (1 4).

• **Automatización e IA:** Ante la escasez de profesionales calificados en respuesta a incidentes, la automatización y la inteligencia artificial se utilizan cada vez más para gestionar tareas rutinarias, permitiendo que los analistas humanos se enfoquen en problemas más complejos.

• **Equipos personalizados:** Los equipos de respuesta a incidentes suelen formarse *ad hoc*, adaptándose a las circunstancias específicas de cada incidente. Este enfoque flexible permite ajustarse a la naturaleza dinámica de las amenazas de seguridad.

METODOLOGÍA DE TRABAJO

La metodología adoptada en este proyecto se fundamenta en el ciclo de diseño de ingeniería, un enfoque iterativo y secuencial ampliamente utilizado en proyectos tecnológicos, que abarca desde la definición conceptual hasta la validación experimental. El propósito central de este proyecto es garantizar un desarrollo ordenado y validado en cada una de sus fases, permitiendo que los resultados obtenidos sean medibles, reproducibles y relevantes para la gestión de incidentes de ciberseguridad. A continuación, se enumerará los pasos principales que se tomarán en el desenlace del proyecto.

1. Revisión bibliográfica y definición conceptual
2. La primera fase consistió en un análisis sistemático de la literatura especializada publicada en los últimos cinco años, con el objetivo de identificar:

* Fortalezas y limitaciones de los sistemas SIEM y SOAR.
* Propuestas recientes de arquitecturas multi-agente basadas en IA.
* Taxonomías de severidad y esquemas de normalización de eventos.

Este análisis permitió diseñar un Esquema Canónico de Eventos (CEC) compuesto por ocho campos esenciales: event\_id, timestamp, source, rule\_id, severity (1–5), asset\_id, confidence (0–1) y raw\_event. El CEC facilita la ingesta de datos heterogéneos y habilita trazabilidad y priorización consistente (Islam, Babar & Nepal, 2019; Castro et al., 2025).

1. Obtención de data

Posteriormente, se planificó la obtención de datas que comprendan escenarios comunes de ciberseguridad, tales como:

* Escaneos de red.
* Ataques de fuerza bruta.
* Intentos de exfiltración de información.

Cada evento será rotulado con un nivel de severidad en una escala de 1 (informativo) a 5 (crítico), validado como verdadero positivo, y documentado en un ground truth que servirá como referencia en la fase de evaluación. Esta estrategia responde a la necesidad de contar con conjuntos de datos confiables y controlados para pruebas reproducibles (Roelofs et al., 2024).

1. Desarrollo incremental de agentes

El sistema se construirá de manera modular, siguiendo un esquema de desarrollo incremental:

* Agente de Monitoreo: encargado de la ingesta y normalización de eventos, implementado en n8n. Su función es mapear los registros al formato CEC y almacenarlos con metadatos de auditoría.
* Agente de Análisis: basado en técnicas de Recuperación Aumentada por Generación (RAG) y Modelos de Lenguaje de Gran Escala (LLMs). Su rol será priorizar incidentes, reducir falsos positivos y generar explicaciones en lenguaje natural para apoyar a analistas humanos (Blefari et al., 2025; Song et al., 2024).
* Agente de Respuesta: encargado de ejecutar playbooks automatizados, que incluirán bloqueo de direcciones IP, cuarentena de endpoints comprometidos, cierre de puertos y apertura de tickets de evidencia (Jiang et al., 2025).

Cada agente será probado y validado en fases sucesivas bajo un enfoque test-driven development (TDD).

1. Integración y orquestación

Una vez construidos los agentes, estos serán integrados en un marco de orquestación utilizando n8n como herramienta central. Posteriormente, se explorará la adopción de LangGraph para mejorar la coordinación entre agentes autónomos y la resiliencia del sistema (Paduraru et al., 2025).

Se implementará un mecanismo de control denominado kill-switch (AUTO\_MODE) que garantizará la reproducibilidad de las pruebas y evitará la ejecución de acciones destructivas en fases preliminares del prototipo.

A continuación, se presenta los pasos de implementación de un playbook orientado a gestionar un ataque de denegación de servicio:

# DDoS Attack Incident Response Playbook

## **Step 1: Identification**

* Monitor network traffic.
* Run the mitigation script (mitigate\_ddos.py).

## **Step 2: Containment**

* Implement rate limiting.
* Block malicious IPs.

## **Step 3: Eradication**

* Update firewall rules.
* Use DDoS protection services.

## **Step 4: Recovery**

* Restore normal traffic flow.
* Monitor for additional attacks.

## **Step 5: Lessons Learned**

* Conduct a post-incident review.
* Update security policies.

1. Evaluación del sistema

El desempeño del orquestador multi-agente será evaluado en un entorno simulado de incidentes, con la finalidad de medir indicadores clave de rendimiento (KPIs) frente a una línea base sin automatización. Los indicadores definidos son:

* Precisión (Accuracy) ≥ 0,80.
* Área bajo la curva (AUC) ≥ 0,80.
* Tasa de falsos positivos (FPR) ≤ 10%.
* Correlación de Kendall (τ) ≥ 0,6 en la priorización de incidentes.
* Reducción del MTTR (Mean Time to Resolution) ≥ 30%.

Adicionalmente, si es posible, se realizarán pruebas de carga y resiliencia, con el fin de verificar el rendimiento en picos de eventos y la capacidad del sistema de mantener operaciones controladas ante la caída de uno o más agentes (Lin et al., 2025).

La metodología descrita busca asegurar que el desarrollo del orquestador multi-agente no solo cumpla los objetivos de reducción de tiempos y falsos positivos, sino que también establezca un marco de validación riguroso, alineado con estándares de reproducibilidad académica y práctica profesional en ciberseguridad.

DESARROLLO DEL PROTOTIPO

Esta etapa busca demostrar la viabilidad técnica de un orquestador multi-agente de IA aplicado a la gestión de incidentes de ciberseguridad.

1. Esquema Canónico de Eventos (CEC)

La base del prototipo es el Esquema Canónico de Eventos (CEC), diseñado para garantizar la normalización de registros de seguridad heterogéneos. El CEC está compuesto por los siguientes campos:

* event\_id: identificador único, generado mediante hash de atributos clave (fuente, regla, timestamp).
* timestamp: registro temporal exacto del evento.
* source: origen del incidente (IDS, firewall, sistema operativo, aplicación).
* rule\_id: regla o firma que disparó la alerta.
* severity: nivel de severidad en una escala de 1 (informativo) a 5 (crítico).
* asset\_id: recurso o activo afectado por el evento.
* confidence: confianza del incidente en una escala de 0 a 1.
* raw\_event: contenido completo del evento en formato original para auditoría.

Este esquema asegura la interoperabilidad y trazabilidad, elementos clave para el análisis forense y la priorización de alertas (Islam, Babar & Nepal, 2019; Roelofs et al., 2024).

A continuación, se presenta un flujograma de la arquitectura preliminar de prototipo del sistema Orquestador en cuestión:

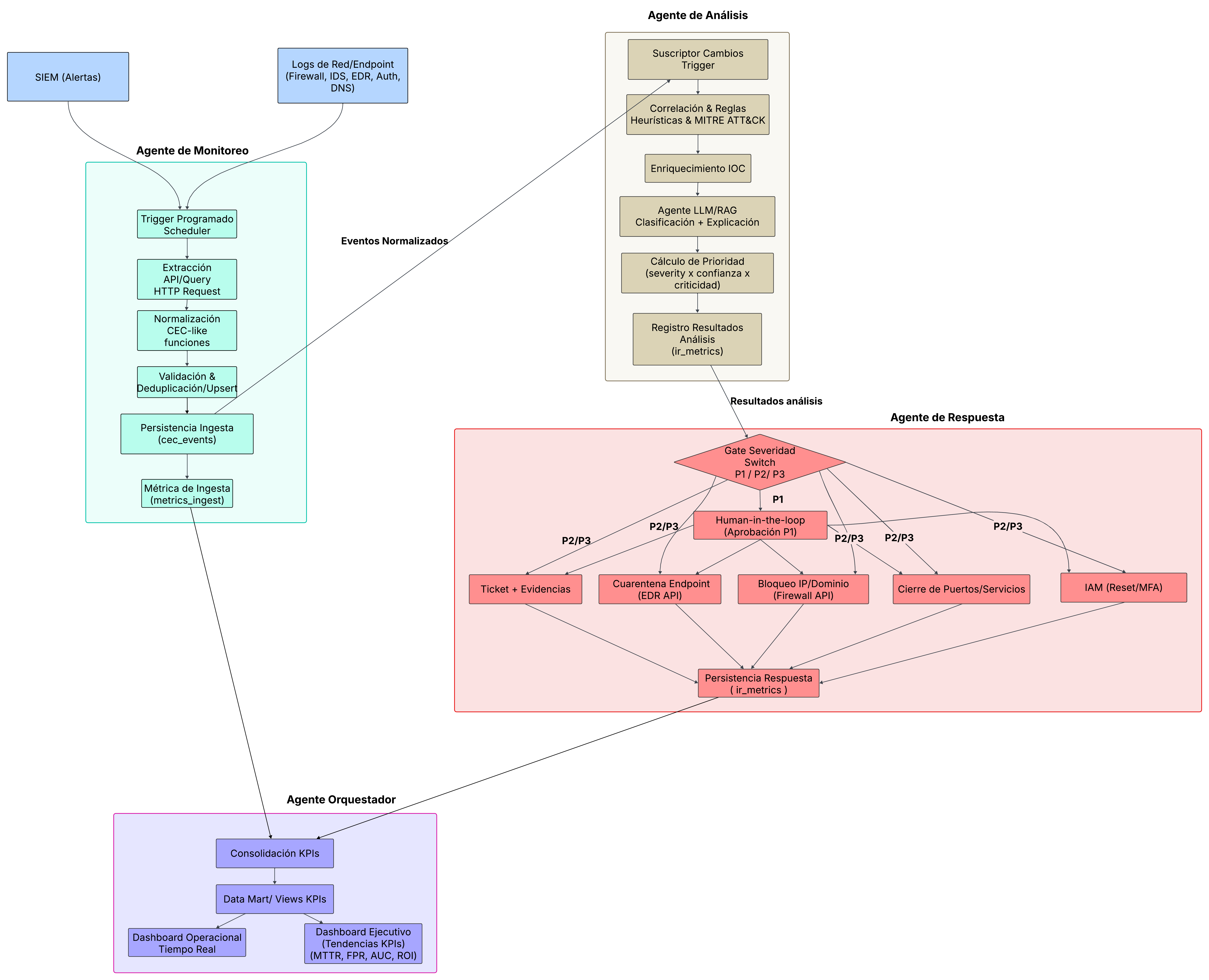


Figura 1. Flujograma Preliminar de Orquestador

2. Agente de Monitoreo en n8n

El primer agente implementado es el de Monitoreo, cuya función principal es la ingesta y normalización de datos en el formato definido por el CEC. Para ello, se utilizó la plataforma de orquestación de flujos n8n, que permite integrar nodos de entrada, transformación y almacenamiento.

El flujo de trabajo diseñado incluye las siguientes etapas:

1. Trigger: ejecución manual o programada (por ejemplo, cada hora).
2. Ingesta de eventos: lectura desde un dataset de pruebas (CSV o API).
3. Transformación: mapeo automático de los campos al formato CEC.
4. Persistencia: almacenamiento de eventos normalizados en base de datos o archivos CSV con metadatos de auditoría.
5. Publicación: envío de los eventos normalizados a una cola lógica o sub-workflow para ser procesados por el agente de análisis.

3. Persistencia y trazabilidad

Cada evento procesado se almacena junto con metadatos de auditoría, lo que permite garantizar:

* Replicabilidad de las pruebas realizadas.
* Análisis forense en caso de incidentes críticos.
* Priorización consistente en las etapas posteriores de análisis y respuesta.

Este enfoque se alinea con prácticas recomendadas en sistemas de seguridad de nueva generación, donde la observabilidad y trazabilidad son fundamentales para auditorías internas y cumplimiento normativo (Kremer et al., 2023).

4. Integración con agentes de análisis y respuesta

Si bien el agente de monitoreo constituye la primera fase del prototipo, este fue diseñado para integrarse con módulos adicionales:

* Agente de Análisis: aplicará clasificación mediante técnicas RAG+LLM, reduciendo falsos positivos y priorizando eventos críticos (Blefari et al., 2025; Song et al., 2024).
* Agente de Respuesta: ejecutará playbooks automatizados, incluyendo bloqueo de direcciones IP, cuarentena de dispositivos y apertura de tickets de investigación (Jiang et al., 2025).

El modularidad de este diseño permite añadir o reemplazar agentes sin comprometer el funcionamiento del sistema completo, siguiendo enfoques de escalabilidad y resiliencia ya probados en literatura (Paduraru et al., 2025). A continuación, se presenta un diagrama de cómo se compone esta integración.

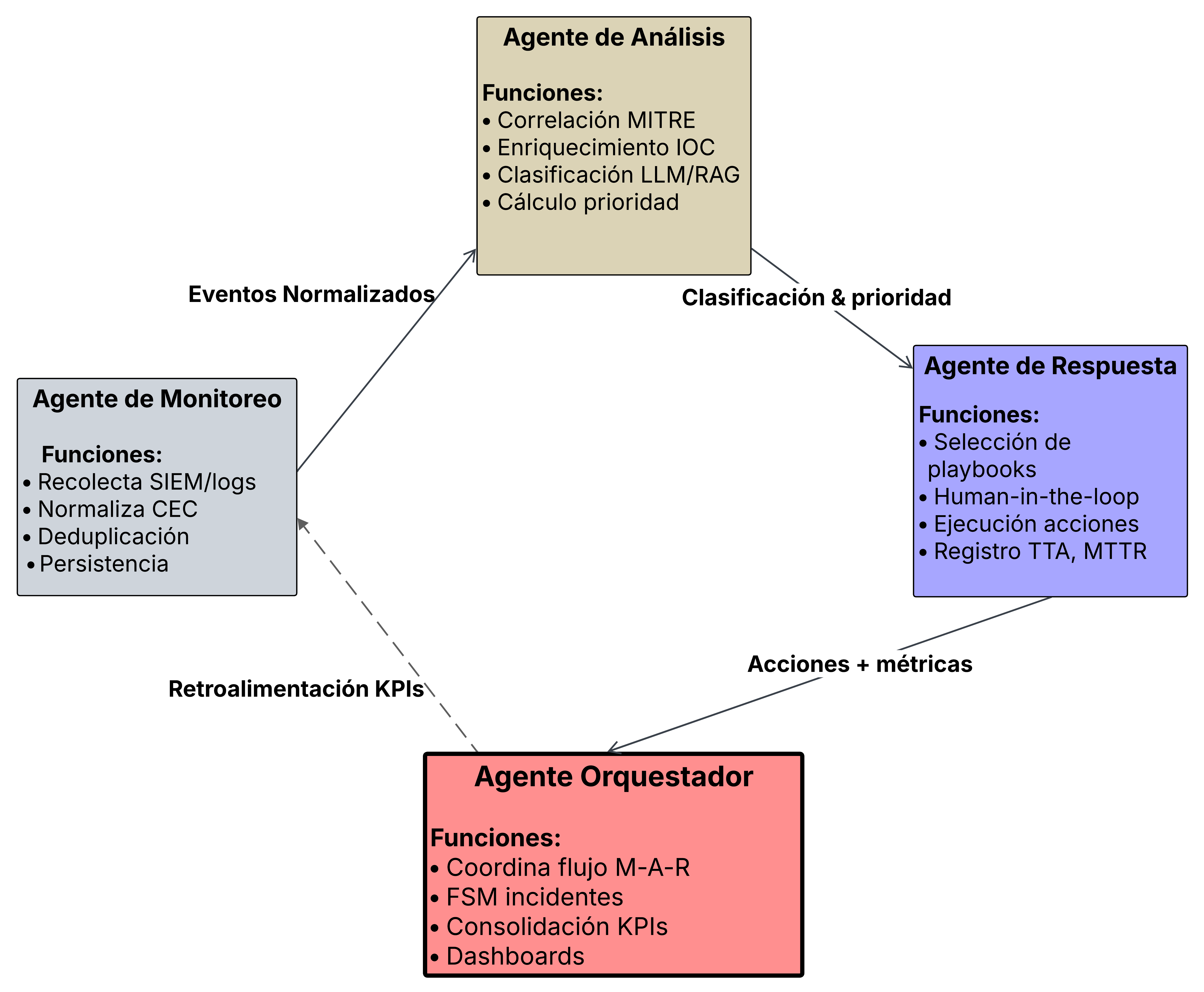


Figura 2. Diagrama preliminar de Arquitectura de agentes

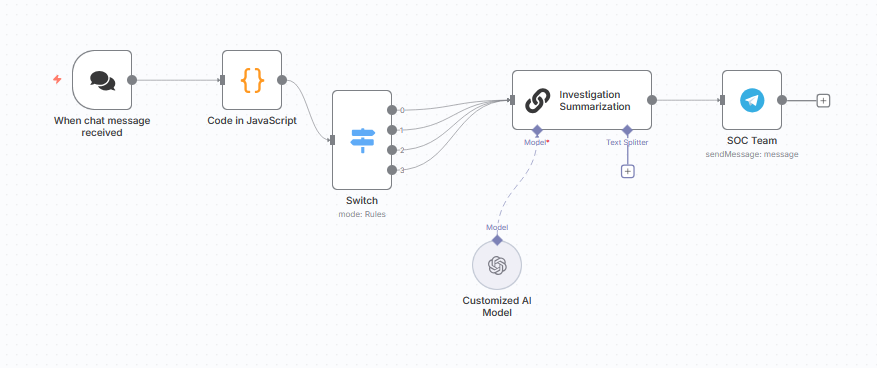
|  |  |  |
| --- | --- | --- |
| **Aspecto** | **En el artículo** | **En nuestra propuesta básica** |
| Uso de Wazuh para detección | Sí — base del artículo | Sí, la idea es recibir alertas de Wazuh |
| Filtrado/análisis adicional | Sí — con ML que clasifica / detecta anomalías | Sí — pero con GPT para “análisis narrativo” y criterio humano |
| Objetivo de disminuir ruido/falsos positivos | Sí, reducir alertas no útiles | Implícito: que GPT ayude a evitar ruido o guiar al SOC |
| Enfoque automático | Sí, el pipeline se automatiza | Sí, workflow automático n8n |
| Velocidad / tiempo real | Sí — sugiere latencias de decenas de ms | Nuestra versión básica es menos rigurosa en latencia (depende del plan del LLM etc.) |

Tabla 4. Comparación en literatura vs propuesta

### Diferencias más importantes

1. **ML vs LLM / heurísticas**
   1. El artículo se basa en modelos de ML clásicos (Random Forest, DBSCAN, etc.), que operan en base de datos / características cuantitativas.
   2. Nosotros estamos usando (o proponiendo usar) un modelo de lenguaje como GPT para análisis narrativo, interpretación de contexto, sugerencias.
   3. Los modelos ML pueden ser más consistentes, rápidos y con menor costo (cuando ya entrenados) para tareas específicas de clasificación binaria/anomalía.
2. **Precisión, tasa de falsos positivos y operacionalidad**
   1. El artículo mide métricas como precisión, recall, tasa de falsos positivos, latencia, escalabilidad, etc.
   2. En nuestra propuesta básica, no hay un ciclo formal de entrenamiento / evaluación; la calidad dependerá del diseño del prompt, del modelo de LLM y de ajustes iterativos.
3. **Requisito de datos y entrenamiento**
   1. Para ML necesitas datos históricos etiquetados, características bien definidas, proceso de entrenamiento / validación.
   2. Para la solución GPT / n8n que proponemos, no necesitas (inicialmente) una base etiquetada, sino que se basa más en interpretación “en vivo” del evento.
4. **Detección de anomalías vs explicación / triage**
   1. El artículo va más hacia la detección automática de anomalías / clasificación.
   2. Lo nuestro es más hacia el **triage asistido**, es decir: “¿es esta alerta digna de atención?”, “¿qué sugerencia damos?”, etc.
5. **Latencia / rendimiento**
   1. En producción, un modelo ML bien optimizado puede responder en pocas decenas de ms. El artículo reporta latencias dentro del umbral real-time.
   2. Usar GPT o un modelo de lenguaje puede tener latencias más altas (depende de la infraestructura, el tráfico, la cola, etc.). Para un entorno muy demandante (miles de eventos por segundo) puede no escalar sin ajustes.
6. **Escalabilidad y costos**
   1. ML bien implementado escala bien “linealmente” para muchos eventos.
   2. Usar GPT frecuentemente (por cada alerta) puede resultar en costos (uso de API, tasa de tokens) y límites operativos.
7. **Interpretabilidad vs caja negra**
   1. Los modelos ML clásicos permiten extraer “feature importance”, comprender qué variables influyeron, etc.
   2. LLM entrega explicación narrativa, pero puede ser más “caja negra” en cuanto a por qué decidió algo (aunque podemos pedir explicaciones en el prompt).

Para abordar la implementación de este prototipo es importante tomar en cuenta el siguiente flujo de trabajo:

Figura 3. Flujo preliminar con webchat hacia SOC

### Desr455555555555555555555555555555555555555555555555555555555555555555555555555555555555555o9cripción del flujo por nodo.

El primer flujo corresponde a la implementación de un agente de monitoreo que recibe datos a través de un nodo de entrada conversacional y los procesa para clasificar y priorizar incidentes de seguridad. Este flujo comienza con el nodo **“When chat message received”**, que actúa como el punto de entrada de datos, permitiendo recibir tanto texto plano como estructuras JSON desde un canal conversacional. Esta flexibilidad es útil para integrar diferentes tipos de entradas, como mensajes humanos o eventos generados por otros sistemas de monitoreo.

Posteriormente, el flujo se dirige al nodo **“Code in JavaScript”**, que contiene la lógica principal de análisis y clasificación. Este bloque ejecuta un script que interpreta los datos entrantes y calcula una serie de métricas relevantes: tasa de paquetes por segundo, número de direcciones IP de origen únicas, duración del evento y nivel de criticidad del activo afectado. A partir de estos valores, el sistema estima una puntuación de riesgo compuesta y determina una prioridad categorizada en cuatro niveles (Low, Medium, High, Critical). Esta etapa incorpora además una bandera denominada *HITL* (Human In The Loop) que marca los casos que requieren revisión humana debido a la baja confianza del modelo o alto impacto potencial.

El resultado de este procesamiento pasa al nodo **“Switch”**, que actúa como un enrutador condicional en función del nivel de prioridad determinado. En este punto se diferencian los caminos según la criticidad del incidente, permitiendo personalizar las acciones posteriores de acuerdo con la gravedad detectada. En todos los casos, el flujo continúa hacia el nodo **“Investigation Summarization”**, que corresponde a un proceso de análisis y síntesis automatizada usando técnicas de *chain summarization* de lenguaje natural. Este nodo, apoyado en el modelo definido en **“Customized AI Model”**, el cual genera un reporte estructurado de investigación con base en las variables de entrada, siguiendo un formato predefinido que incluye nombre y descripción de la alerta, táctica y técnica MITRE, alcance, reputación de artefactos externos y recomendaciones de seguridad.

El informe producido se envía mediante el nodo **“SOC Team”**, encargado de la comunicación directa con el equipo de operaciones de seguridad a través de Telegram. Este paso permite asegurar una distribución inmediata del resumen de incidentes hacia el canal operativo, garantizando que la información crítica llegue a los analistas en tiempo real. Finalmente, el flujo contempla un nodo **“Create Spreadsheet”**, diseñado para registrar de manera opcional los resultados en Google Sheets, permitiendo mantener una trazabilidad histórica de los incidentes procesados.

En conjunto, este flujo funciona como un agente de monitoreo inteligente capaz de recibir entradas dinámicas, calcular niveles de riesgo de forma autónoma, resumir la información mediante un modelo de lenguaje y distribuir los resultados a un canal de comunicación operativo. La integración de procesamiento numérico, clasificación heurística y lenguaje natural permite automatizar tareas que tradicionalmente requerían intervención manual, optimizando la velocidad de respuesta ante alertas de seguridad.

A continuación, se presenta una captura de la compilación de este flujo con el siguiente mensaje de entrada:

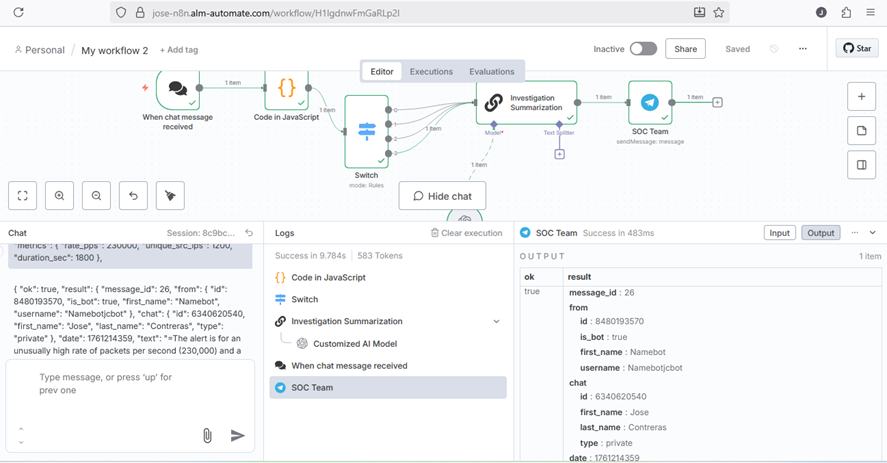
"metrics": {

"rate\_pps": 230000,

"unique\_src\_ips": 1200,

"duration\_sec": 1800

},

Figura 4. Compilación flujo con mensaje prueba

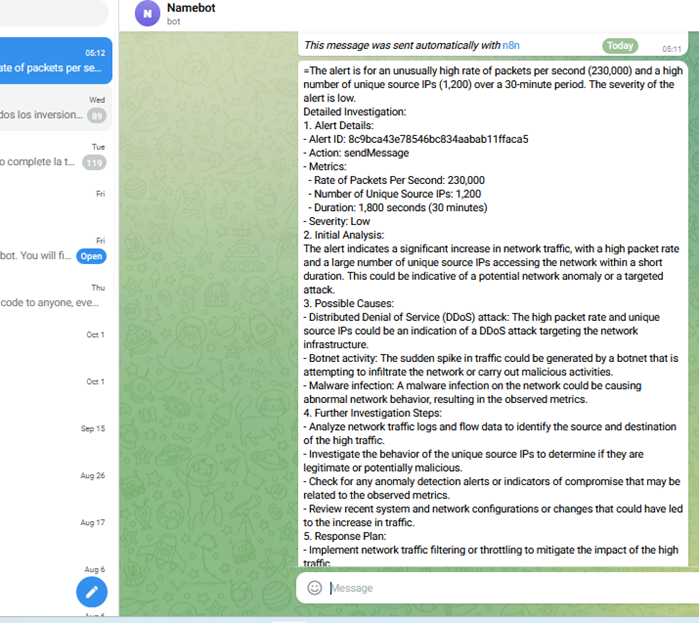
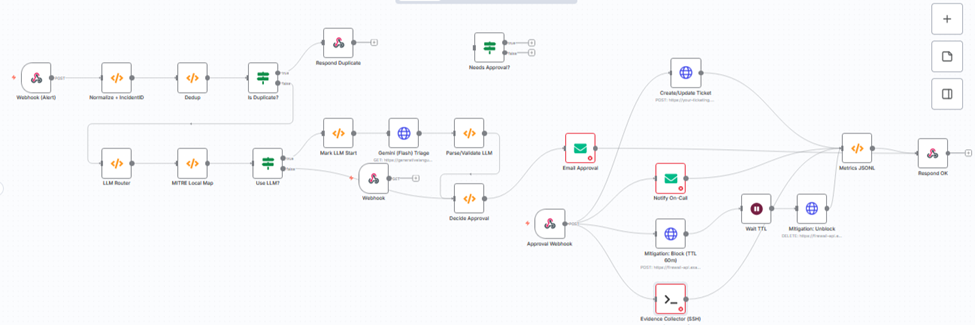


Figura 5. Mensaje de salida Telegram SOC

Teniendo en cuenta el diagrama de arquitectura presentado en el agente de monitoreo y se procede a describir el mismo:

Figura 6. Flujo de monitoreo con Wazuh alert

**Descripción del flujo por nodo:**

El flujo de monitoreo desarrollado tiene como objetivo automatizar la recepción, análisis y priorización de eventos de seguridad provenientes del entorno del SIEM Wazuh. Este flujo se basa en una estructura modular de nodos que trabajan de manera secuencial para gestionar de forma eficiente las alertas generadas por el sistema.

El proceso inicia con la recepción de los eventos a través de un webhook que actúa como punto de entrada, seguido de la normalización de los datos y la generación de un identificador único por incidente. Posteriormente, se aplica un mecanismo de deduplicación que evita procesar alertas repetidas y, dependiendo de su complejidad, las envía a análisis mediante un modelo de lenguaje avanzado o a un camino de mitigación directa. Este flujo permite integrar técnicas del marco MITRE ATT&CK, generando reportes de triage y ejecutando acciones automáticas como el bloqueo temporal de direcciones IP o la notificación a equipos de respuesta. En conjunto, este diseño permite una gestión integral del ciclo de monitoreo y respuesta ante incidentes, asegurando trazabilidad, eficiencia y una reducción significativa del tiempo medio de detección y mitigación (MTTD/MTTR).

A continuación, se resume el flujo anterior a nivel de nodo:

|  |  |  |
| --- | --- | --- |
| **Imagen** | **Nodo** | **Descripción** |
|  | **Normalize + IncidentID** | Este nodo recibe la alerta cruda y la normaliza. Extrae datos clave como dirección IP, aplicación afectada y severidad, y genera un identificador único de incidente (*IncidentID*) calculado a partir de estos campos y una ventana temporal de 5 minutos. Este ID permite asegurar la trazabilidad y evitar duplicidades en los eventos registrados. |
|  | **Dedup** | Implementa un mecanismo de deduplicación temporal basado en memoria estática. Si un *IncidentID* ya ha sido procesado recientemente (dentro de los últimos cinco minutos), marca la alerta como duplicada y evita su reprocesamiento, optimizando así el rendimiento del flujo. |
|  | **Respond Duplicate** | Cuando el nodo anterior detecta una alerta repetida, este bloque responde directamente al webhook inicial, deteniendo la ejecución del flujo. De esta manera, se evita el consumo innecesario de recursos computacionales en incidentes ya gestionados. |
|  | **MITRE Local Map** | Asocia automáticamente la alerta con técnicas y sub-técnicas del marco MITRE ATT&CK. Lo hace a partir de palabras clave detectadas en la descripción o el tráfico (por ejemplo, “HTTP”, “SYN”, “UDP”), enriqueciendo el contexto del incidente con información de tácticas y técnicas reconocidas. |
|  | **Parse/Validate LLM** | Este nodo se encarga de interpretar y validar la respuesta JSON generada por el modelo de lenguaje (*Gemini 1.5 Flash*). Extrae información de triage, plan de acción y mapeo MITRE, y asegura que el resultado cumpla con el esquema esperado antes de continuar con la automatización. |
|  | **Decide Approval** | Analiza la información resultante del modelo de lenguaje y determina si el plan de mitigación requiere revisión humana. Si el caso es crítico o ambiguo, marca el proceso como “pendiente de aprobación”, asegurando que las acciones sensibles sean revisadas por un analista. |
|  | **Approval Webhook** | Recibe las aprobaciones humanas mediante una petición HTTP (POST). Este nodo actúa como puerta de enlace entre el analista y el flujo automatizado, permitiendo habilitar o denegar acciones de mitigación en tiempo real. |
|  | **Create/Update Ticket** | Crea o actualiza un ticket en el sistema de gestión de incidentes (como Jira, ServiceNow o Zendesk). Registra los detalles del evento, el plan de acción y las referencias MITRE, garantizando la trazabilidad administrativa del proceso. |
|  | **Mitigation: Block (TTL 60m)** | Envía una solicitud POST a la API del firewall para bloquear la dirección IP atacante durante un tiempo limitado (TTL de 60 minutos). Este paso representa la ejecución automática de una contramedida técnica frente a ataques de denegación de servicio. |
|  | **Respond OK** | Es el nodo final del flujo. Envía una respuesta al sistema que generó la alerta, confirmando que el incidente fue procesado correctamente y que las acciones de mitigación o registro fueron completadas sin errores. |
|  | **Webhook (Alert)** | Es el punto de entrada del flujo. Recibe las alertas mediante solicitudes HTTP tipo POST provenientes de sistemas externos o del SIEM. Su función es capturar los datos de eventos DoS (Denial of Service) para iniciar el proceso automatizado de análisis y mitigación. |
|  | **Is Duplicate?** | Evalúa si la alerta actual ya fue procesada recientemente. Si detecta que el incidente es duplicado, redirige la ejecución hacia el nodo *Respond Duplicate*; si no lo es, continúa hacia el análisis de clasificación. |
|  | **LLM Router** | Determina si el evento será procesado por la vía rápida (*fast path*) o si se requiere una intervención más compleja con modelo de lenguaje (*use\_llm*). Para ello, analiza métricas del tráfico y palabras clave, clasificando las alertas obvias o ambiguas. |
|  | **Use LLM?** | Nodo condicional que bifurca el flujo dependiendo del valor del parámetro *use\_llm*. Si es verdadero, activa el análisis con el modelo de lenguaje; si es falso, continúa con las acciones predefinidas de mitigación. |
|  | **Mark LLM Start** | Registra la marca temporal en la que comienza el procesamiento del modelo de lenguaje. Este tiempo se usa posteriormente para calcular métricas de rendimiento del sistema y latencia del análisis. |
|  | **Gemini (Flash) Triage** | Envía los datos de la alerta al modelo *Gemini 1.5 Flash* mediante una solicitud HTTP. El modelo analiza el contexto del incidente y genera un informe estructurado con triage, plan de acción, técnicas MITRE y posibles indicadores de falso positivo. |
|  | **Wait TTL** | Introduce una pausa controlada equivalente al tiempo de vida del bloqueo temporal aplicado (TTL = 60 minutos). Su función es garantizar que la medida de mitigación permanezca activa el tiempo necesario antes de revertirse. |
|  | **Metrics JSONL** | Registra información operativa del flujo, como el tiempo total de ejecución, uso del modelo LLM, duración de bloqueos, nivel de prioridad y estadísticas de precisión. Esta información alimenta las métricas de rendimiento y auditoría del sistema. |

Tabla 5. Descripción Flujo de Monitoreo por nodo

Este flujo demuestra una arquitectura de respuesta automatizada altamente integrada, capaz de combinar reglas lógicas, modelos de lenguaje, comunicación asíncrona y operaciones de red en un solo ciclo operativo. La inclusión del modelo *Gemini* permite mejorar la interpretación contextual de las alertas, mientras que los mecanismos de deduplicación, aprobación y reversión garantizan seguridad y control operacional. En conjunto, este flujo consolida las capacidades de orquestación y respuesta del sistema, alineándose con los principios de un *Security Orchestration, Automation and Response (SOAR)* moderno.

En ambos flujos se puede observar un enfoque modular y jerárquico en la gestión de incidentes de ciberseguridad. El primer flujo enfatiza la etapa de **detección y análisis**, centrada en la evaluación heurística y la priorización automática, mientras que el segundo aborda la **respuesta y mitigación** mediante la integración de modelos generativos y automatización de infraestructura. La combinación de ambas arquitecturas representa un ciclo completo de monitoreo inteligente: desde la identificación inicial del riesgo hasta la ejecución de medidas concretas de contención y registro de incidentes.

Este diseño evidencia un uso estratégico de tecnologías de inteligencia artificial y automatización de procesos, donde los modelos de lenguaje natural no reemplazan al analista humano, sino que amplifican su capacidad operativa al sintetizar, clasificar y contextualizar información compleja en tiempo real. En el entorno experimental del laboratorio de ciberseguridad descrito, estos flujos permiten recrear y gestionar ataques controlados (como DDoS o ransomware), midiendo la eficacia de las respuestas y generando datos valiosos para la mejora continua del sistema.

Se procedió a la construcción de un nuevo flujo de trabajo a partir de la estructura preliminar.

1. Levantamiento de Laboratorio:

El levantamiento del laboratorio de ciberseguridad se llevó a cabo con el propósito de construir un entorno controlado para la simulación de ataques y la validación del funcionamiento del orquestador. Para ello, se implementó un servidor Wazuh Manager en una máquina virtual preconfigurada con sistema Wazuh v4.13.1 OVA. A continuación, se detalla el procedimiento:

1. Actualización del sistema base:

Sudo apt update && sudo apt upgrade -y

1. Activación de servicios:

sudo systemctl start wazuh-manager  
 sudo systemctl enable wazuh-manager  
 sudo systemctl status wazuh-manager

1. Acceso a la interfaz web del Dashboard:

Dentro del navegador de preferencia, se direcciona hacia la IP donde se encuentra alojado el servidor. En este caso:

https://<192.168.100.98>

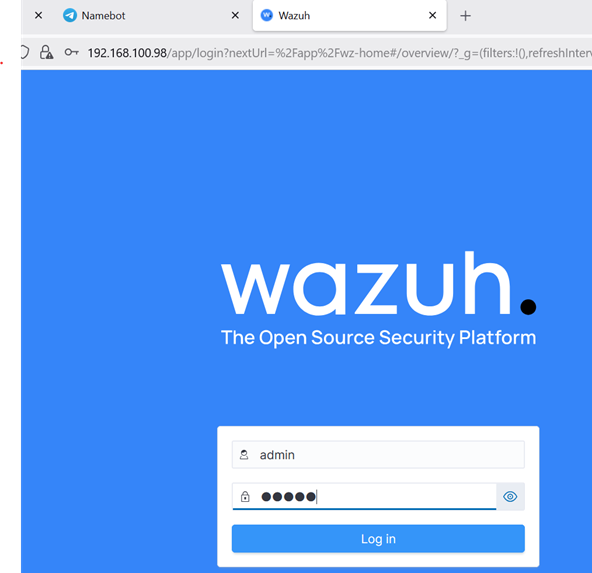


Figura 7. Acceso a Dashboard

Posteriormente, el dashboard procederá a cargarse presentando los datos actuales al momento:

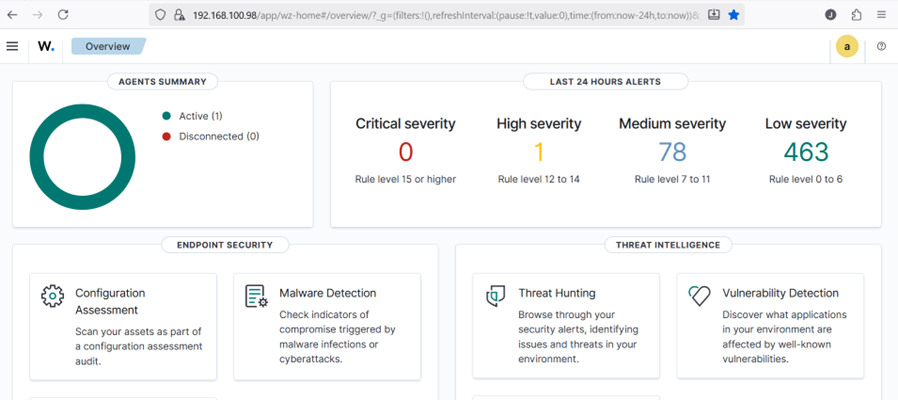


Figura 8. Dashboard Wazuh

Una vez dentro es importante verificar los detalles de conexión del endpoint, inicialmente se procedió con una máquina con Windows 11, entonces seleccionando en el recuadro de Agent Summary a los equipos activos se tiene:

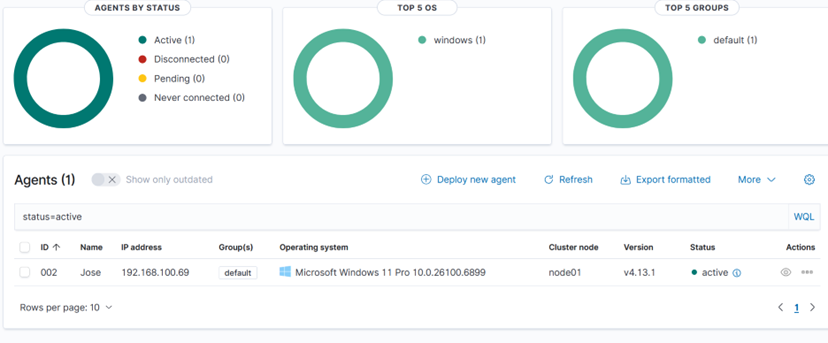


Figura 9. Detalle de conexión Endpoint

Finalmente, se verifica que la IP descrita sea igual a la de la máquina del usuario:

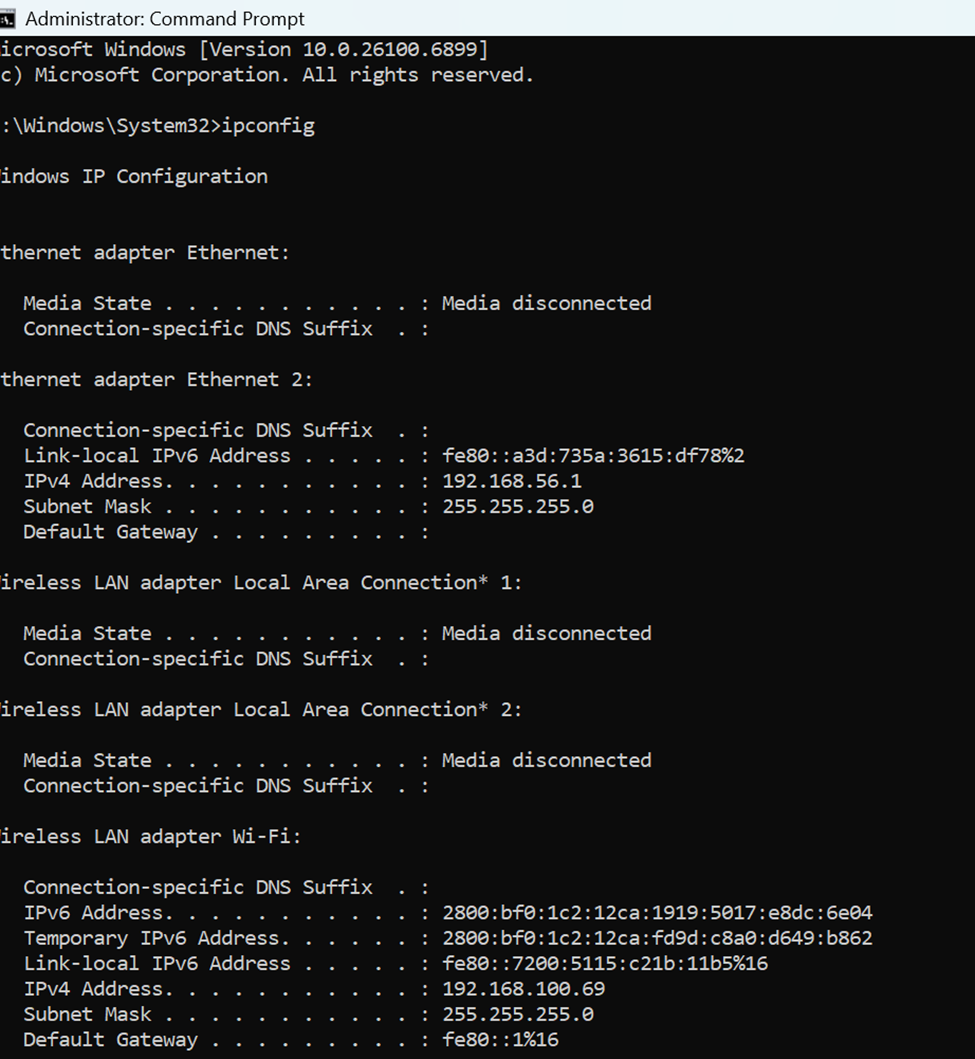


Figura 10. IP de máquina cliente

1. Configuración de agentes simulados:

Una vez iniciado sesión en el servidor, se entra con derechos de administrador a la herramienta para manejar los agentes:

sudo /var/ossec/bin/manage\_agents

Se procede a seleccionar la opción A,

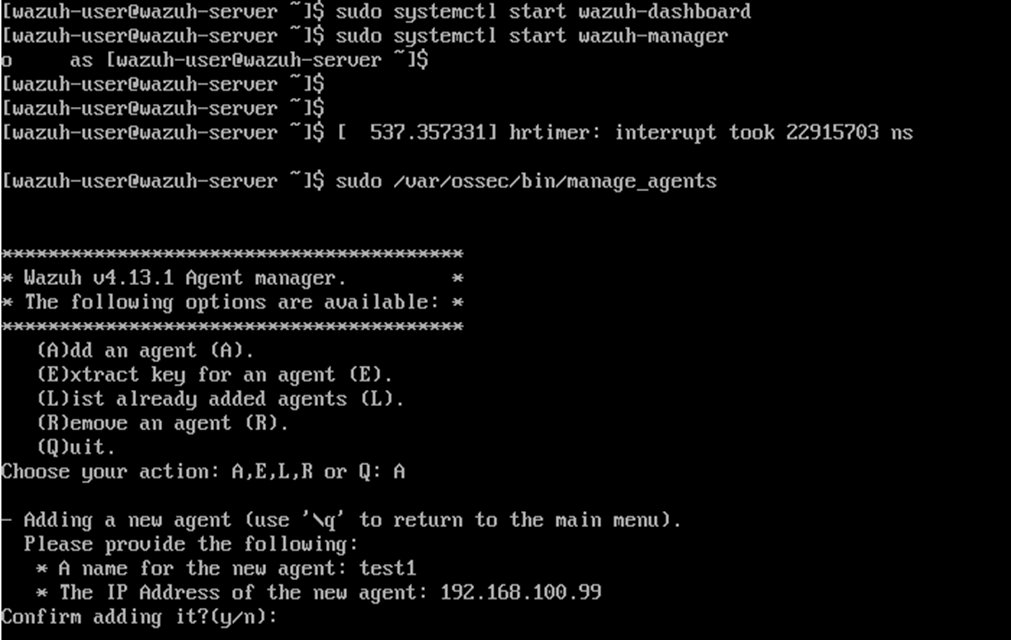
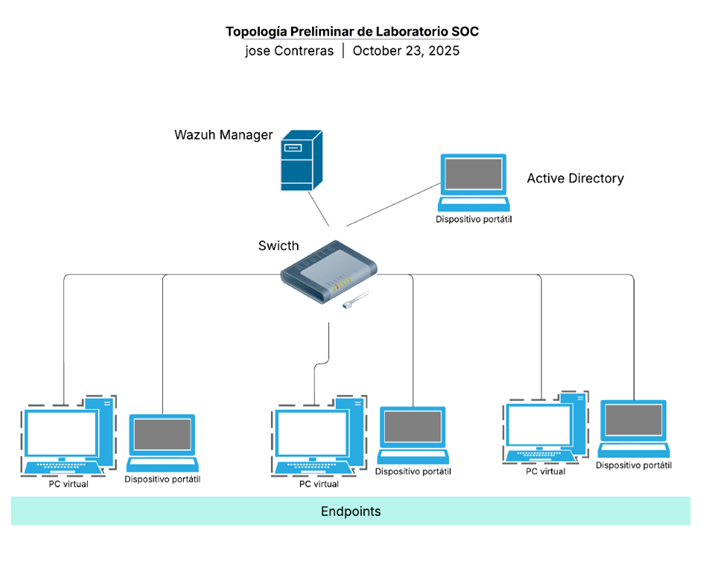


Figura 11. Listado de agentes

Para luego ingresar un alias, y su respectiva dirección IP con el fin de que pueda ser reconocida por el servidor. Finalmente, se reinicia el servicio wazuh-agent para refrescar los agentes.

sudo systemctl restart wazuh-agent

Con esto, el servidor se encuentra levantado y la máquina cliente enlazada, información que se encontrará reflejada en el dashboard y servidor. A continuación, se muestra un diagrama con la topología a implementarse de forma completa:

Figura 12. Topología de Laboratorio SOC

**Descripción de diagrama**

La topología representada en la figura muestra la arquitectura preliminar del laboratorio SOC implementado para la validación del proyecto. En ella se puede observar un servidor Wazuh Manager que actúa como núcleo del sistema de monitoreo y gestión de eventos de seguridad (SIEM), encargado de centralizar la recopilación de logs, correlacionar alertas y coordinar la comunicación con los agentes distribuidos en los diferentes equipos finales. Conectado a este se encuentra un servidor de Active Directory, utilizado para administrar usuarios, políticas de autenticación y control de acceso en el entorno experimental. Ambos servidores están interconectados a través de un switch de red, que constituye el punto de enlace principal entre los distintos componentes del laboratorio.

En el nivel inferior se ubican los endpoints, conformados por una combinación de PCs virtuales y dispositivos portátiles donde se instalan los agentes Wazuh, los cuales envían eventos de seguridad al servidor principal. Esta estructura permite recrear escenarios de ataque controlado, como intentos de autenticación fallidos, accesos no autorizados o patrones de tráfico asociados a ataques DDoS, proporcionando un entorno funcional para el análisis de alertas y la evaluación de las capacidades de detección y respuesta del sistema. En conjunto, esta topología reproduce las condiciones de un Centro de Operaciones de Seguridad (SOC), integrando de manera práctica los procesos de monitoreo, análisis y respuesta automatizada en un ambiente controlado de laboratorio.

# Conclusiones

Presenta los aportes de este trabajo con base en lo investigado, es importante que como autor puedas analizar el tema y su relevancia para la profesión dentro del contexto nacional e internacional (presenta similitudes, diferencias entre los diferentes enfoques del tema investigado). En el caso de presentaciones artísticas o creativas se debe describir de qué se tratan y justificar sus elementos, obligatoriamente incluir anexos con fotos, evidencias (partituras, enlaces a videos, etc.) del producto elaborado. Realiza un análisis de lo que has aprendido en este trabajo, incluye sugerencias de estudios posibles que se realicen en el futuro para comprender de mejor manera el tema, menciona alguna dificultad que hayas tenido para realizar este trabajo y sus razones.

# Referencias bibliográficas

Alshamrani, A. (2025). Federated hierarchical MARL for zero-shot cyber defense. PLoS ONE. https://doi.org/10.1371/journal.pone.0329969

Blefari, F., Cosentino, C., Pironti, F. A., Furfaro, A., & Marozzo, F. (2025). CyberRAG: An agentic RAG cyber attack classification and reporting tool. arXiv. https://doi.org/10.48550/arXiv.2507.02424

Brahmandam, B. A. (2025). AI driven ChatOps for DevSecOps: Automating security incident response. International Journal of Multidisciplinary Research in Science, Engineering and Technology, 8(2), 85–96. https://doi.org/10.15680/ijmrset.2025.0802085

Castro, S. R., Campbell, R., Lau, N., Villalobos, O., Duan, J., & Cardenas, A. A. (2025). Large language models are autonomous cyber defenders. Conference on Algebraic Informatics. https://doi.org/10.1109/CAI64502.2025.00195

European Union Agency for Cybersecurity (ENISA). (2023). ENISA threat landscape 2023. Publications Office of the European Union. https://www.enisa.europa.eu/publications

Islam, C., Babar, M. A., & Nepal, S. (2019). Automated interpretation and integration of security tools using semantic knowledge. In Advanced Information Systems Engineering (pp. 529–544). Springer. https://doi.org/10.1007/978-3-030-21290-2\_32

Jiang, Y., Oo, N., Meng, Q., Lin, L., Niyato, D., Xiong, Z., Lim, H. W., & Sikdar, B. (2025). CyGATE: Game-theoretic cyber attack-defense engine for patch strategy optimization. IEEE. https://doi.org/10.1109/CSR61664.2024.10679456

Kremer, R., Wudali, P. N., Momiyama, S., Araki, J., Furukawa, J., Elovici, Y., & Shabtai, A. (2023). IC-SECURE: Intelligent system for assisting security experts in generating playbooks for automated incident response. arXiv. https://doi.org/10.48550/arXiv.2311.03825

Lin, X., Zhang, J., Deng, G., Liu, T., Liu, X., Yang, C., Guo, Q., & Chen, R. (2025). IRCopilot: Automated incident response with large language models. arXiv. https://doi.org/10.48550/arXiv.2505.11901

Nyberg, J., & Johnson, P. (2024). Structural generalization in autonomous cyber incident response with message-passing neural networks and reinforcement learning. Computer Science Symposium in Russia. https://doi.org/10.1109/CSR61664.2024.10679456

Paduraru, C., Patilea, C., & Stefanescu, A. (2025). CyberGuardian 2: Integrating LLMs and agentic AI assistants for securing distributed networks. International Conference on Evaluation of Novel Approaches to Software Engineering. https://doi.org/10.5220/0013406000003928

Roelofs, T.-M., Bárbaro, E., Pekarskikh, S., Orzechowska, K., Kwapien, M., Tyrlik, J., Smadu, D., van Eeten, M., & Zhauniarovich, Y. (2024). Finding harmony in the noise: Blending security alerts for attack detection. ACM Symposium on Applied Computing. https://doi.org/10.1145/3605098.3635981

Song, C., Ma, L., Zheng, J., Liao, J., Kuang, H., & Yang, L. (2024). Audit-LLM: Multi-agent collaboration for log-based insider threat detection. arXiv. https://doi.org/10.48550/arXiv.2408.08902

Tsinghua University, Georgia Tech, & Microsoft. (2025). Triangle: Empowering incident triage with Multi-LLM-Agents. arXiv. https://doi.org/10.48550/arXiv.2505.20945

U.S. Cybersecurity and Infrastructure Security Agency (CISA). (2024). CISA cyber incidents report 2024. U.S. Department of Homeland Security. https://www.cisa.gov

# Anexo A: Título

(Si hubiere anexos. Ver requerimientos de anexos obligatorios para trabajos de carreras relacionadas a las artes)

# Anexo B: Título

(Si hubiere anexos. Ver requerimientos de anexos obligatorios para trabajos de carreras relacionadas a las artes)

# Anexo C: Título

(Si hubiere anexos. Ver requerimientos de anexos obligatorios para trabajos de carreras relacionadas a las artes)

Se recomienda iniciar cada anexo en una nueva hoja. Se puede incluir anexos adicionales (ANEXO D: TÍTULO, ANEXO E: TÍTULO, ANEXO F: TÍTULO, etc.) de conforme la necesidad de presentación de los mismos en el trabajo. Ver requerimientos de anexos obligatorios para trabajos de carreras relacionadas a las artes.