UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

TRABAJO DE INTEGRACIÓN CURRICULAR

IMPLEMENTACIÓN DE SENSORES IDS PARA EL MONITOREO Y LA SEGURIDAD EN EL LABORATORIO DE TELECOMUNICACIONES

AUTOR

RICARDO VILLAO GREGORY EZEQUIEL

PREVIO A LA OBTENCIÓN DEL GRADO ACADÉMICO EN
INGENIERO EN TELECOMUNICACIONES

TUTOR

ING. LUIS MIGUEL AMAYA FARIÑO Mgt.

Santa Elena, Ecuador

Año 2025

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

Ing. Ronald Rovira Jurado, PhD
DIRECTOR DE LA CARRERA

Ing. Luis Miguel Amaya Fariño, Mgt.
TUTOR

Ing. Fernando V. Chamba Macas, MSc.
DOCENTE ESPECIALISTA

Ing. Luis Miguel Amaya Fariño, Mgt
DOCENTE GUÍA UIC

Ing. Corina Gonzabay De La A, Mgt
SECRETARIA

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA

FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que luego de haber dirigido científica y técnicamente el desarrollo y estructura final del trabajo, este cumple y se ajusta a los estándares académicos, razón por el cual apruebo en todas sus partes el presente trabajo de titulación que fue realizado en su totalidad por **Gregory Ezequiel Ricardo Villao**, como requerimiento para la obtención del título de Ingeniero en Telecomunicaciones.

La Libertad, a los 17 días del mes de diciembre del año 2025

TUTOR

Ing. Luis Miguel Amaya Fariño, Mgt.

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA

FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Gregory Ezequiel Ricardo Villao**

DECLARO QUE:

El trabajo de Titulación, **Implementación De Sensores IDS Para El Monitoreo Y La Seguridad En El Laboratorio De Telecomunicaciones** previo a la obtención del título en Ingeniero en Telecomunicaciones, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

La Libertad, a los 17 días del mes de diciembre del año 2025

EL AUTOR

Gregory Ezequiel Ricardo Villao

UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

AUTORIZACIÓN

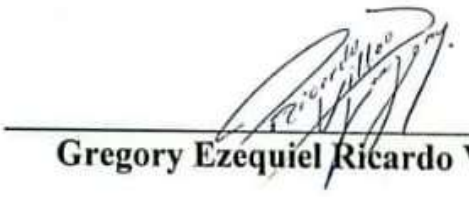
Yo, **Gregory Ezequiel Ricardo Villao**

Autorizo a la Universidad Estatal Península de Santa Elena, para que haga de este trabajo de titulación o parte de él, un documento disponible para su lectura consulta y procesos de investigación, según las normas de la Institución.

Cedo los derechos en línea patrimoniales de artículo profesional de alto nivel con fines de difusión pública, además apruebo la reproducción de este artículo académico dentro de las regulaciones de la Universidad, siempre y cuando esta reproducción no suponga una ganancia económica y se realice respetando mis derechos de autor

Santa Elena, a los 17 días del mes de diciembre del año 2025

EL AUTOR



Gregory Ezequiel Ricardo Villao

AGRADECIMIENTO

Expreso mi sincero agradecimiento a Dios por la fortaleza, constancia y oportunidades que me permitieron culminar esta etapa académica.

A mi familia, por su apoyo permanente a lo largo de este proceso. A mis padres Jimmy y Maricruz; a mis hermanos Irinés, Jimmy y Marinés; a mi sobrino Austin y a mi cuñado Luis Pozo por su acompañamiento, comprensión y palabras de aliento.

A mis abuelos paternos, Mami Jovi y Papi Roso, y a mis abuelos maternos, Mi Martita y Papá Gollo, por su amor y guía constante.

A mis tíos paternos Ing. Iván Daniel, Lic. Melissa y al Lic. Iván Muñoz, Alexandra y Fabian; así como a mis tíos maternos Evelyn, Jonathan, Carlos y Nancy, a mis primos Lic. Ivanna, Psi. Dana Daniela, Ing. Apolito, Lic. Melina, Ián, Jovita, Ing. Jair, Tlgo. Kenneth, Erick, Jonás y Josué por su apoyo y cercanía durante este camino académico.

A la familia Rivera Alcívar, especialmente a la señora Verónica y al señor Jorge Abel, junto a Annabell, Aitana, Esteban, Andrés, Pamela, Farid y Doménica, por el respaldo brindado y por formar parte importante de mi entorno familiar.

A mis amigos dentro de la universidad, Joselyn, Bruce, Freddy Marmolejo, Joan Ramirez, Michael Moreira y Génesis Panchana, con quienes inicié esta etapa y a quienes fortalecieron la amistad hasta el final de la carrera: Jhonny, Sebastián, Heidy, Maeba, Andy, Carmen, Peralta, Gabriel, Erick Yagual, Henry, José Daniel y Anthony Rosales, por el compañerismo y apoyo mutuo.

A mis buenos amigos y colegas actuales Cristhian, Anthony Lascano, Dilan, Julio, Rubén, Gaby, Mafer, Kenjy, Joshua y Jonathan González quienes fueron testigos directos de este proceso y brindaron su apoyo constante.

Finalmente, a mis mejores amigos Karelys, Iván, Chineco Joffre Iván, Helen y Darwin, por sus años de amistad sincera, leal, divertida y de acompañamiento incondicional.

Gregory Ezequiel Ricardo Villao

DEDICATORIA

Dedico este trabajo con todo mi amor y gratitud a mi familia, pilar fundamental de mi vida y fuente constante de fortaleza.

A mi hija Gianna Mariel mi mayor bendición y la razón más profunda de cada esfuerzo realizado. Tu existencia me enseñó el verdadero significado de la responsabilidad, la perseverancia y el deseo de ser mejor cada día.

A su mamá, Yelena Rivera por su apoyo y comprensión en momentos clave de este camino.

A mis padres, Jimmy Ricardo y Maricruz Villao por su amor incondicional, por creer en mí incluso en los momentos de mayor dificultad y por enseñarme con su ejemplo el valor del sacrificio, la disciplina y la honestidad.

A mis abuelos paternos, Mami Jovi y Papi Roso y a mis abuelos maternos Mi Martita y Papá Gollo quienes con su sabiduría, consejos y ejemplo de vida sembraron en mí principios que hoy me sostienen como persona y profesional.

Este logro es el reflejo de todo lo que me han brindado. A ustedes que son mi raíz y mi motivo, dedico este esfuerzo.

Gregory Ezequiel, Ricardo Villao

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente proyecto implementa un Sistema de Detección de Intrusos (IDS) en el Laboratorio de Telecomunicaciones mediante una arquitectura híbrida compuesta por un router MikroTik CRS310, los motores de análisis Suricata y Zeek, y un sistema de observabilidad basado en Loki y Promtail. El método aplicado fue experimental, desarrollándose un entorno controlado donde se configuraron sensores IDS en RouterOS y se integraron herramientas de recolección y procesamiento de logs para monitorear el tráfico en tiempo real.

Los datos generados por Suricata y Zeek fueron centralizados mediante Promtail hacia Loki, permitiendo su visualización en un Dashboard de Grafana. Durante las pruebas se generaron diferentes patrones de tráfico, logrando identificar 62 alertas de Suricata y 47 eventos relevantes registrados por Zeek, alcanzando una precisión promedio del 93 % en la clasificación de actividades sospechosas. Además, se desarrolló un sistema de notificaciones automatizadas mediante Telegram, lo que redujo el tiempo de respuesta ante incidentes a menos de un minuto.

Los resultados demuestran que la integración de estas herramientas permite fortalecer la seguridad del laboratorio, proporciona un monitoreo continuo y facilita la toma de decisiones mediante paneles centralizados y alertas en tiempo real.

Palabras clave: IDS, Suricata, Zeek, MikroTik, Grafana, Loki, Promtail.

ABSTRACT

This project implements an Intrusion Detection System (IDS) in the Telecommunications Laboratory through a hybrid architecture composed of a MikroTik CRS310 router, the Suricata and Zeek analysis engines, and an observability system based on Loki and Promtail. The applied method was experimental, developing a controlled environment where IDS sensors were configured in RouterOS and log collection tools were integrated to monitor traffic in real time.

The data generated by Suricata and Zeek were centralized through Promtail into Loki, enabling visualization in a Grafana dashboard. During testing, different traffic patterns were generated, allowing the identification of 62 Suricata alerts and 47 relevant events recorded by Zeek, achieving an average accuracy of 93% in classifying suspicious activities. In addition, an automated Telegram notification system was developed, reducing incident response time to less than one minute.

The results demonstrate that the integration of these tools strengthens laboratory security, provides continuous monitoring, and facilitates decision-making through centralized panels and real-time alerts.

Keywords: IDS, Suricata, Zeek, MikroTik, Grafana, Loki, Promtail.

INTRODUCCIÓN

El aumento de incidentes de ciberseguridad en entornos académicos ha evidenciado la necesidad de implementar mecanismos que permitan supervisar el tráfico de red y reaccionar oportunamente ante actividades sospechosas. Diversos estudios señalan que los ataques actuales presentan mayor complejidad y requieren sistemas capaces de analizar patrones en tiempo real [1], [2]. En laboratorios de telecomunicaciones donde convergen dispositivos heterogéneos y prácticas experimentales estas vulnerabilidades se intensifican y pueden afectar la continuidad de actividades docentes y de investigación.

Los Sistemas de Detección de Intrusos (IDS) basados en red se han consolidado como herramientas esenciales para identificar patrones sospechosos y prevenir accesos no autorizados [3], [4]. En este contexto se indica que el uso de dispositivos MikroTik y la integración de plataformas como Grafana permiten desarrollar soluciones accesibles, escalables y eficientes. De este modo la implementación de un IDS junto con alertas automatizadas y visualización centralizada representa una estrategia clave para fortalecer la seguridad del laboratorio y mejorar la capacidad de respuesta ante incidentes [5], [6].

El propósito de este estudio es implementar un IDS funcional que detecte y visualice eventos en tiempo real fortaleciendo la seguridad del laboratorio. El trabajo se delimita al análisis la configuración y evaluación de sensores de detección de intrusos sin incluir técnicas ofensivas ni ataques avanzados.

CAPÍTULO I

1.1 Identificación del problema

El rápido avance tecnológico y el crecimiento de las redes de telecomunicaciones han provocado un aumento significativo de las ciberamenazas, especialmente en entornos académicos como los laboratorios de telecomunicaciones. Estos laboratorios manejan datos sensibles y equipos de prueba, lo que los convierte en objetivos vulnerables de ataques como escaneos de puertos, inyecciones de payload y ataques de denegación de servicio (DoS) [7], [8]. Es así como muchos laboratorios carecen de soluciones de seguridad efectivas que permitan detectar y mitigar estas amenazas en tiempo real.

Los Sistemas de Detección de Intrusos (IDS) son herramientas clave para identificar actividades sospechosas y prevenir accesos no autorizados. Sin embargo, muchos laboratorios no implementan IDS actualizados ni herramientas de visualización integradas, lo que aumenta la exposición a riesgos operativos y afecta la continuidad de las investigaciones[9]. Esta situación destaca la necesidad de contar con un sistema robusto que pueda monitorear el tráfico de red y generar alertas automáticas ante la detección de actividades anómalas[2].

La causa principal de la vulnerabilidad de los laboratorios de telecomunicaciones radica en la falta de soluciones de monitoreo en tiempo real. Sin un IDS adecuado, las amenazas pasan desapercibidas lo que puede llevar a pérdidas de datos críticos interrupciones en las investigaciones o incluso daños a la infraestructura.

Los efectos de la ausencia de monitoreo eficaz son inmediatos ya que las redes académicas al ser compartidas por diversos dispositivos y usuarios son objetivos fáciles para los ciberataques.

La falta de un sistema de alertas automatizadas como un bot de Telegram que agrava aún más la situación ya que retrasa la reacción ante incidentes.

1.2. Antecedentes

El desarrollo de los Sistemas de Detección de Intrusos ha evolucionado significativamente durante las últimas décadas pasando de enfoques basados únicamente en firmas a modelos capaces de analizar comportamiento, patrones estadísticos y correlación de eventos [3][4]. Aunque tecnologías recientes aprovechan Machine Learning y Deep Learning para mejorar la precisión de la detección [10][11] los entornos educativos requieren soluciones eficientes de bajo costo y fáciles de integrar a infraestructuras existentes.

Estudios como los de [12] y [1] destacan la importancia de los IDS basados en red (NIDS) para entornos donde el tráfico fluye a través de un punto central como routers de distribución. Estos sistemas permiten detectar patrones de ataque, escaneos, anomalías y accesos no autorizados sin la necesidad de instalar agentes en cada equipo.

En semejante las herramientas de visualización como lo es Grafana han ganado popularidad debido a su capacidad para representar métricas de seguridad en tiempo real esto facilitando la toma de decisiones y el análisis operacional [5][13]. Asimismo, los sistemas de alertas automatizadas mediante mensajería digital son hoy considerados esenciales para acortar el tiempo de respuesta ante incidentes [6], [14].

La combinación de estas tecnologías permite desarrollar soluciones efectivas y accesibles, especialmente en laboratorios donde el monitoreo continuo es clave para evitar interrupciones y garantizar la seguridad del entorno académico.

1.3. Descripción del Proyecto

El presente proyecto implementa un Sistema de Detección de Intrusos (IDS) utilizando un router MikroTik CRS310 con RouterOS como plataforma central. A través de reglas de inspección y filtrado para cada puerto es configurado como un punto de monitoreo las cuales son los sensores IDS capaz de identificar patrones de tráfico inusual y detectar amenazas como escaneos, intentos de acceso no autorizado o comportamientos anómalos.

Al mismo tiempo se desarrolla un sistema automatizado que genera notificaciones en tiempo real hacia un canal definido por el administrador y esto permitiendo que cualquier actividad sospechosa sea atendida de inmediato. Para complementar estas funciones se diseña un Dashboard en Grafana que consolida métricas, alertas y registros de eventos así facilitando la interpretación del estado de la red y apoyando el análisis forense posterior.

El proyecto se adapta a las necesidades del Laboratorio de Telecomunicaciones ofreciendo una solución práctica, escalable y alineada a la realidad operativa de instituciones educativas donde no siempre es posible desplegar infraestructura de seguridad compleja.

1.4. Objetivos del proyecto

Objetivo General:

Implementar un sistema de detección de intrusos que proporcione una solución de seguridad integral para el laboratorio de telecomunicaciones mejorando la capacidad de la red para detectar y responder a posibles ataques y amenazas cibernéticas.

Objetivos Específicos:

- Configurar un Router MikroTik con Sensores IDS estableciendo un entorno seguro y controlado donde cada puerto del router MikroTik funcione como un punto de monitoreo, detectando y analizando el tráfico de red en tiempo real.

- Diseñar e implementar un Dashboard en Grafana que permita visualizar en tiempo real las alertas, métricas de tráfico y eventos generados por el sistema IDS para el monitoreo y la respuesta ante incidentes en el Laboratorio de Telecomunicaciones.

- Desarrollar un Sistema de Notificación Automatizado con la creación de un script que envíe alertas por Telegram al personal encargado cada vez que se detecte una actividad sospechosa o un posible evento maligno.

1.5. Justificación

El fortalecimiento de la seguridad en redes académicas se ha convertido en una prioridad debido al aumento significativo de ataques que buscan explotar vulnerabilidades en entornos donde convergen múltiples dispositivos y usuarios. Investigaciones recientes muestran que las intrusiones dirigidas a instituciones educativas han incrementado su complejidad, superando la capacidad de los mecanismos tradicionales de monitoreo [2], [15]. En laboratorios de telecomunicaciones donde se realizan prácticas con tráfico real y configuraciones experimentales estas amenazas se intensifican esto poniendo en riesgo la integridad, disponibilidad y continuidad de los procesos formativos.

La implementación de Sistemas de Detección de Intrusos (IDS) se presenta como una solución viable y efectiva para identificar anomalías y patrones de ataque antes de que se materialicen en incidentes de seguridad. Los IDS basados en red (NIDS) son particularmente adecuados para este entorno ya que permiten supervisar el tráfico desde un punto central sin requerir agentes adicionales en cada equipo esto favoreciendo su adopción en laboratorios universitarios [3], [12]. Además los estudios demuestran que el uso de análisis de comportamiento y correlación de eventos mejora la capacidad de detectar actividades sospechosas incluso en redes pequeñas o medianas [1], [16].

El uso de MikroTik como plataforma para la implementación del IDS resulta especialmente pertinente ya que su sistema operativo RouterOS permite crear reglas avanzadas y esto monitorear paquetes en tiempo real automatizando respuestas ante comportamientos anómalos. Su accesibilidad económica y flexibilidad lo convierten en una herramienta apropiada para instituciones con recursos limitados [15],[17]. Y es así como la integración de visualizadores como

Grafana potencia la interpretación de métricas dando facilidad de la correlación de eventos y la toma de decisiones rápidas mediante paneles dinámicos [5], [13].

Un componente fundamental en los sistemas de seguridad modernos es la notificación inmediata de incidentes. Estas investigaciones recientes respaldan el uso de bots de mensajería como los es la aplicación Telegram para alertar de manera efectiva al personal responsable reduciendo el tiempo de reacción y fortaleciendo la capacidad de respuesta [6], [14].

En conjunto la implementación de un IDS basado en MikroTik complementado con un sistema automatizado de alertas y un Dashboard de visualización constituye una solución integral, económica y formativa que contribuye significativamente al fortalecimiento de la postura de seguridad del Laboratorio de Telecomunicaciones.

1.6.Alcance del Proyecto

El presente proyecto abarca el diseño, configuración e implementación de un Sistema de Detección de Intrusos (IDS) basado en un router MikroTik CRS310 dentro del Laboratorio de Telecomunicaciones. El alcance comprende la creación de sensores IDS utilizando reglas específicas en RouterOS para analizar el tráfico en tiempo real, así como la inclusión de un mecanismo automatizado que genere alertas inmediatas cuando se detecten comportamientos anómalos o intentos de intrusión. Este sistema se complementa con la elaboración de un Dashboard en Grafana que consolida métricas, eventos y patrones del tráfico, permitiendo una visualización clara y dinámica del estado de la red.

El proyecto se limita a la implementación de funciones de monitoreo, detección y alerta dentro de la red interna del laboratorio, utilizando únicamente la infraestructura disponible. No se contempla el uso de técnicas de ataque ofensivo, equipos especializados para pruebas de

penetración ni la integración de sistemas externos de ciberseguridad. Tampoco se desarrollan mecanismos de respuesta automática distintos a la generación de alertas.

Finalmente, el alcance incluye pruebas funcionales con tráfico real generado por los propios dispositivos del laboratorio, con el fin de validar el correcto desempeño del IDS y la precisión en la detección de actividades sospechosas.

1.7. Marco contextual

Este proyecto tiene su raíz en el Laboratorio de Telecomunicaciones de la Facultad de Sistemas y Telecomunicaciones de la Universidad Estatal Península de Santa Elena (UPSE), situado en La Libertad, Ecuador, un espacio académico vibrante dedicado a la investigación y la capacitación en tecnologías de redes. Equipado con una infraestructura de red de punta, este laboratorio sirve como un refugio para estudiantes e investigadores que trabajan con datos confidenciales y equipos de gran valor, como routers y herramientas de prueba, en experimentos que impulsan el conocimiento. Sin embargo, el creciente realce en las redes lo hace vulnerable a ciberamenazas, destacando la urgencia de un Sistema de Detección de Intrusos (IDS) para proteger este entorno.

El IDS se construye alrededor de los routers MikroTik CRS310-1G-5S-4S+IN y RB4011iGS+5HacQ2HnD-IN cuya adaptabilidad y economía los convierten en aliados ideales para entornos académicos con recursos ajustados. Las pruebas se enriquecerán con el Hak5 Pineapple WiFi, simulando ataques en una red controlada con VLAN 10. Este esfuerzo, anclado en las necesidades de seguridad de la región costera de Ecuador, no solo busca robustecer la infraestructura de red, sino también fomentar la formación en ciberseguridad, dejando una huella positiva en el desarrollo académico y tecnológico local.

CAPÍTULO II

2.1.Marco Conceptual

El marco conceptual establece los fundamentos necesarios para comprender la implementación de un Sistema de Detección de Intrusos (IDS) en el Laboratorio de Telecomunicaciones de la Universidad Estatal Península de Santa Elena. Se abordan los conceptos esenciales de ciberseguridad, los tipos de IDS y la relevancia de las herramientas empleadas en este proyecto.

2.1.1 Necesidad de un Sistema de Detección de Intrusos y las Ciberamenazas

El incremento de ataques dirigidos a infraestructuras educativas y redes de investigación ha sido ampliamente documentado en estudios recientes, los cuales destacan un crecimiento sostenido en incidentes como escaneos, accesos no autorizados y ataques de denegación de servicio [16]. Estas amenazas afectan especialmente a laboratorios que operan con tráfico real y dispositivos heterogéneo.

Frente a este escenario, los IDS se consolidan como mecanismos fundamentales para identificar patrones anómalos y detectar actividades maliciosas en tiempo real [7]. Su implementación en redes académicas permite mejorar la visibilidad del tráfico y reducir la exposición a riesgos operativos.

2.1.2 Rol del Sistema de Detección de Intrusos en Entornos Académicos

En laboratorios la dinámica experimental y el constante intercambio de datos incrementan la superficie de ataque. Un IDS contribuye a supervisar la red registrar eventos relevantes y generar alertas oportunas esto favoreciendo la continuidad de actividades académicas y prácticas de laboratorio [16].

El uso de motores de análisis como Suricata y Zeek mejora la profundidad del monitoreo, ya que permiten identificar firmas conocidas y analizar comportamientos, respectivamente[9], [13]. Su integración con plataformas de observabilidad como Loki–Promtail y paneles visuales como Grafana facilita la interpretación de métricas y el análisis de tendencias [8].

2.2. Marco Teórico

2.2.1 Conceptos Básicos de Sistemas de Detección de Intrusos

Los (IDS) sistemas de detección de intrusos son instrumentos de la ciberseguridad bosquejadas para monitorear el tráfico de la red, tal y como se muestra en la *Figura 1* el comportamiento del sistema en busca de acciones sospechosas y maliciosas. Operan como un sistema de alarma que identifica la penetración, como los intentos de acceso no autorizados o la liberación de servicios (dos), dando una respuesta rápida a la reducción de los riesgos. La identificación consta de sensores para la recopilación de datos, análisis del motor para el procesamiento de información y los mecanismos de alarma para notificar a los administradores [18].

Los IDS bloquea automáticamente la amenaza a diferencia del IPS que son los Sistemas de Penetración pero las detecta y los inspecciona. En el entorno académico como los laboratorios de telecomunicaciones la identificación es importante para proteger datos confidenciales durante los experimentos. Según un estudio reciente el uso de IDS ha reducido los agujeros de seguridad en un 40% en redes complejas [19].

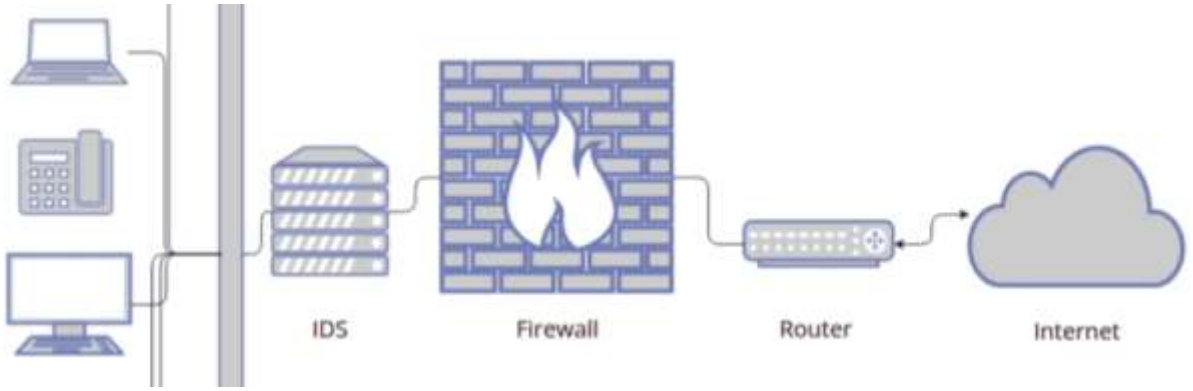


Figura 1 - Funcionalidad de un IDS

2.2.2 Tipos de Sistema de Detección de Intrusos

Los Sistemas de Detección de Intrusos (IDS) Se clasifican de acuerdo con sus métodos de ubicación y monitoreo y se adaptan a las necesidades de seguridad ambiental como laboratorios de telecomunicaciones. Los tipos principales son:

- **Basados en Host (HIDS):** Monitorean un dispositivo específico, analizan registros, procesos y archivos para determinar amenazas internas como Malware o acceso no autorizado. Son ideales para servidores críticos, pero cada host debe instalarse [20].
- **Basados en Red (NIDS):** Analizan el tráfico de la red en puntos estratégicos, como enrutadores para identificar modelos maliciosos, como escaneos de puertos o ataque de servicio (dos). En relación con el proyecto NID ha configurado MikroTik CS310-1G-5S-4S 1N 1N 1N Router Key para proteger la red de laboratorio [21].
- **Híbridos:** Combinan (HIDS) golpes y revestimientos anchos (NIDS) que son útiles en un entorno mixto con múltiples dispositivos conectados [22].
- **Inalámbricos:** Están diseñados para redes Wi-Fi y detectar amenazas como el ataque man-in-the-middle (MITM) [23].

En la siguiente *Tabla 1* es una clasificación permite seleccionar el tipo de IDS adecuado siendo el NIDS el más relevante para el proyecto debido a su implementación en el router MikroTik, que centraliza el monitoreo del tráfico y facilita pruebas con el Hak5 Pineapple WiFi.

Tabla 1 – Tipos de IDS y su aplicación en el proyecto

TIPO DE IDS	UBICACIÓN	APLICACIÓN EN EL PROYECTO
HIDS	Dispositivo Individual	Protección de servidores locales
NIDS	Puntos de Red Router MikroTik	Monitoreo del tráfico general del laboratorio
Hibrido	Dispositivos y red	Cobertura amplia en entornos mixtos
Inalámbrico	Redes Wi-Fi	Detección de ataques simulados

2.2.3 Basados en Host (HIDS)

Los Sistemas de Detección de Intrusos Basados en Host (HIDS) se instalan directamente en dispositivos individuales como servidores o estaciones de trabajo, para monitorear actividades locales en tiempo real. En la *Figura2* analizan logs de sistema, procesos y cambios en archivos, detectando amenazas internas que podrían pasar desapercibidas en la red como un programa maligno o accesos no autorizados[24]. En un laboratorio de telecomunicaciones los HIDS protegen equipos críticos durante experimentos, ofreciendo una vigilancia detallada sin depender de la conectividad de red [23].

Aunque efectivos para amenazas localizadas, los HIDS consumen recursos del host y requieren mantenimiento en cada dispositivo, lo que puede ser desafiante en entornos con múltiples máquinas[25]. Sin embargo, su capacidad para operar offline los hace ideales para escenarios aislados, complementando soluciones de red como el router MikroTik en el proyecto.

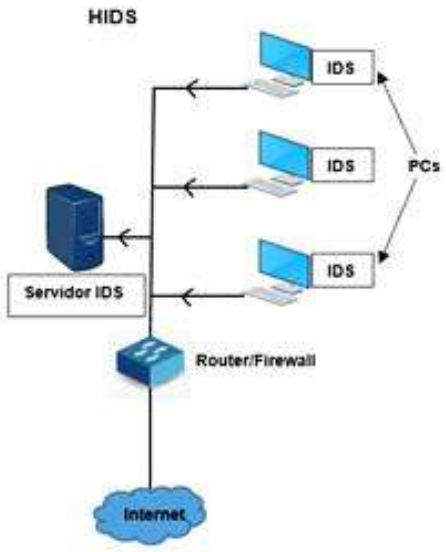


Figura 2 – Comportamiento de un HIDS en una estructura de Red con IDS

- **HIDS Basados en Firmas (SIDS)**

Utilizan patrones predefinidos o "firmas" de amenazas conocidas para identificar intrusiones, comparando actividades del host con una base de datos de malware o comportamientos maliciosos. Esta aproximación ofrece alta precisión y velocidad en la detección de ataques comunes, como virus o exploits específicos, minimizando falsos positivos en entornos controlados como laboratorios.

No obstante los SIDS son limitados contra amenazas nuevas o zero-day ya que dependen de actualizaciones frecuentes de la base de firmas para pruebas que permiten validar su efectividad contra ataques simulados, fortaleciendo la seguridad del host sin sobrecargar recursos.

- **HIDS Basados en Anomalías (HADS)**

Establecen un perfil de comportamiento normal del host mediante aprendizaje automático y alertan sobre desviaciones, como procesos inusuales o accesos irregulares. Esta técnica es

poderosa para detectar amenazas desconocidas, adaptándose a patrones dinámicos en entornos académicos donde se realizan experimentos variados.[20]

A pesar de su versatilidad los HADS pueden generar más falsos positivos debido a variaciones legítimas en el uso del host, requiriendo afinación continua. En el proyecto combinar HADS con el router MikroTik mejora la detección integral optimizando la respuesta a anomalías en tiempo real[21].

2.2.4 Basados en Red (NIDS)

Los Sistemas de Detección de Intrusos Basados en Red (NIDS) monitorean el tráfico de red en puntos estratégicos, como routers o switches, para identificar actividades maliciosas en tiempo real. Analizan paquetes de datos, detectando amenazas como escaneos de puertos, ataques de denegación de servicio (DoS) o intentos de acceso no autorizado, lo que los hace ideales para proteger redes compartidas en laboratorios de telecomunicaciones. En el proyecto, un NIDS configurado en el router MikroTik CRS310-1G-5S-4S+1N utiliza reglas de firewall de RouterOS para vigilar el tráfico, complementado con scripts para alertas automáticas[26].

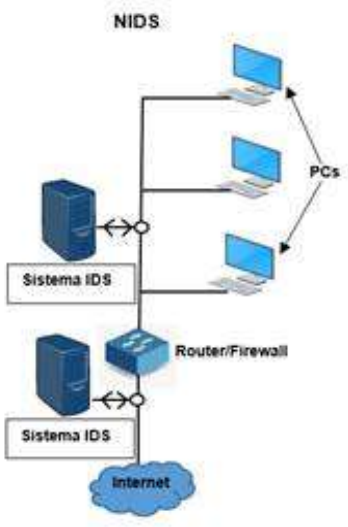


Figura 3 – Comportamiento de un NIDS en una estructura de Red con IDS

Como se muestra en la *Figura3* su capacidad para cubrir múltiples dispositivos desde un solo punto los hace escalables y eficientes. Estos también requieren una ubicación estratégica para maximizar la visibilidad del tráfico[27] .

- **Ventajas Y Desventajas de los NIDS**

Los NIDS ofrecen ventajas significativas como el monitoreo centralizado de toda la red, lo que reduce la carga en dispositivos individuales. En un laboratorio, permiten detectar amenazas externas, como ataques MITM, sin necesidad de instalar software en cada host, integrándose fácilmente con el MikroTik CRS310 para un análisis eficiente del tráfico [28].

Ilustración 1 - Ventajas y Desventajas de NIDS

En la *Ilustración 1* podemos observar condiciones que común mente presentan, también mencionar que su escalabilidad los hace ideales para redes con múltiples dispositivos [29] y su capacidad de detección en tiempo real apoya respuestas rápidas, como las alertas por correo configuradas en el proyecto.

Los NIDS enfrentan limitaciones no pueden analizar tráfico cifrado, lo que dificulta detectar amenazas ocultas, y dependen de la ubicación del sensor, pudiendo pasar por alto tráfico en segmentos de red no monitoreados. También generan falsos positivos en entornos con tráfico variable, requiriendo ajustes en las reglas para optimizar la precisión [23].

2.2.5 Métricas de Evaluación de un IDS

La evaluación de un Sistema de Detección de Intrusos (IDS) depende de métricas que miden su precisión y eficacia en entornos como laboratorios de telecomunicaciones. En el proyecto, las métricas permiten optimizar el NIDS implementado en el router MikroTik CRS310-1G-5S-4S+1N [28].

- **Matriz de Confusión**

La matriz de confusión clasifica los resultados del IDS en cuatro categorías esto indica que en la *Tabla 2* podemos observar cómo existe un orden para la comprensión de esta herramienta verdaderos positivos (TP) que es la detección correcta de amenazas, verdaderos negativos (TN) que es la no detección de tráfico normal, falsos positivos (FP) que es las alertas erróneas y falsos negativos (FN) que son las amenazas no detectadas. En el proyecto, esta matriz evalúa la precisión del NIDS en el MikroTik frente a ataques como MITM [23].

Tabla 2 - Clasificación de la Matriz de Confusión

	Predicción Positiva (Ataque)	Predicción Negativa (Normal)
Realidad Positiva (Ataque)	Verdadero Positivo (TP)	Falso Negativo (FN)
Realidad Negativa (Normal)	Falso Positivo (FP)	Verdadero Negativo (TN)

En un paradigma un FP ocurre si el IDS marca tráfico legítimo como sospechoso, mientras que un FN implica no detectar un ataque real. La matriz permite ajustar las reglas de detección en RouterOS para mejorar la precisión reduciendo FP [30].

En esta matriz se pueden calcular las métricas antes mencionadas [31], usando las siguientes formulas a detallar:

- Tasa de detección:

$$TP\ /\ (TP\ +\ FN)$$

- Tasa de falsos positivos:

$$FP\ /\ (FP\ +\ TN)$$

- Tasa de falsos negativos:

$$FN\ /\ (TP\ +\ FN)$$

La matriz de confusión y la curva ROC (Receiver Operating Characteristic) permiten visualizar el rendimiento y comparar configuraciones o reglas de detección[32].

2.2.6 Conceptos Básicos de Seguridad en Redes

La seguridad en redes protege la confidencialidad, integridad y disponibilidad de datos en entornos conectados, como laboratorios de telecomunicaciones. Incluye medidas para prevenir accesos no autorizados y mitigar amenazas, asegurando que las redes funcionen de manera confiable. En el proyecto estos conceptos guían la implementación del IDS en el router MikroTik para detectar intrusiones en tiempo real [22]. La defensa en profundidad combina múltiples capas de seguridad desde el firewall hasta monitoreo continúo reduciendo riesgos [23].

- **Principios Básicos - Triada CIA**

La triada CIA es la confidencialidad, integridad y disponibilidad, la cual forma el núcleo de la seguridad en redes. La confidencialidad asegura que solo usuarios autorizados accedan a la información, mientras que la integridad previene alteraciones no autorizadas. La disponibilidad garantiza que los recursos estén accesibles cuando se necesitan [20].

Tabla 3 - Triada CIA / Relación con la seguridad de redes

Vulnerabilidad	Capa OSI	Ejemplo de ataque
Spoofing	Enlace	Suplantación de MAC
IP Spoofing	Red	Enrutamiento malicioso
SYN Flood	Transporte	Saturación de servidores
Injection	Aplicación	Manipulación de datos

Las amenazas incluyen ataques que hacen explotar vulnerabilidades para comprometer redes. En laboratorios las amenazas internas como errores humanos pueden introducir riesgos mientras que externas como MITM afectan el tráfico WiFi [22]. El proyecto aborda estas amenazas mediante el NIDS en MikroTik reduciendo la exposición a malware y asegurando una respuesta rápida para mantener la disponibilidad [23].

- **Estrategias de Defensa en Profundidad**

La defensa en profundidad utiliza múltiples capas, como autenticación, cifrado y monitoreo, para crear barreras contra amenazas. En redes combina firewalls con IDS para una protección integral [20]. En el contexto del proyecto esta estrategia integra el IDS en RouterOS

con alertas automatizadas, fortaleciendo la defensa contra ataques simulados y promoviendo la resiliencia en el laboratorio de telecomunicaciones [21].

2.2.7 MikroTik como plataforma IDS, RouterOS y Firewall

Los equipos MikroTik como el router CRS310-1G-5S-4S+1N son dispositivos de red versátiles y rentables esto es diseñado para entornos como laboratorios de telecomunicaciones. Su sistema operativo RouterOS permite configuraciones avanzadas incluyendo firewall y detección de intrusos ideales para implementar un NIDS en el proyecto[17].

MikroTik ofrece flexibilidad para monitorear tráfico y generar alertas adaptándose a las necesidades del laboratorio. Su capacidad de integración lo hacen adecuado para entornos académicos donde el IDS en RouterOS detecta ataques como MITM fortaleciendo la seguridad [33].

- **RouterOS y sus Características: Un Sistema Operativo Versátil**

RouterOS es el sistema operativo de MikroTik que gestiona funciones de red como enrutamiento, firewall y scripting[14]. En el proyecto, permite configurar reglas de detección para el NIDS, analizando tráfico en tiempo real. Su interfaz gráfica y comandos CLI facilitan ajustes precisos [33]. Sus características incluyen soporte para VPN, QoS y monitoreo avanzado esto son esencial para proteger el laboratorio.

El RouterOS es mucho más que un simple sistema operativo para enrutadores. Es una plataforma completa que ofrece una amplia gama de funcionalidades para gestionar redes de todo tipo desde pequeñas redes domésticas hasta grandes infraestructuras empresariales. Su flexibilidad se debe a su arquitectura modular donde permite agregar y quitar funcionalidades según las

necesidades específicas de cada usuario. RouterOS cuenta con una comunidad de usuarios muy activa que desarrolla y comparte scripts, paquetes y configuraciones personalizadas.

- **Licencias MikroTik: Adaptadas a tus Necesidades**

Las licencias de MikroTik están diseñadas para adaptarse a diferentes tamaños de redes y presupuestos. Las licencias de nivel básico ofrecen funcionalidades esenciales para pequeñas redes, mientras que las licencias de nivel empresarial incluyen características avanzadas como balanceo de carga, alta disponibilidad y QoS. Es importante elegir la licencia adecuada para optimizar el rendimiento y la seguridad de tu red [34].

- **Interfaz de Usuario: Flexibilidad y Potencia**

Los RouterOS ofrece múltiples opciones para administrar tus dispositivos:

- ✓ Winbox: Una interfaz gráfica de usuario "GUI" intuitiva que facilita la configuración de los dispositivos especialmente para usuarios principiantes.
- ✓ Línea de comandos: Para usuarios más experimentados la línea de comandos ofrece un mayor control y flexibilidad.
- ✓ API: Una interfaz de programación de aplicaciones que permite automatizar tareas y integrar RouterOS con otros sistemas.

- **Componentes Clave y su Función**

El MikroTik CRS310-1G-5S-4S+1N cuenta con puertos Gigabit, SFP y un procesador robusto, soportando alto volumen de tráfico. Sus componentes clave, como el firewall y el sistema de scripts, son fundamentales para el NIDS en el proyecto, monitoreando ataques simulados [33].

La siguiente *Tabla 4* describe cualitativamente los componentes principales y sus funciones asegurando una implementación efectiva del IDS en el laboratorio.

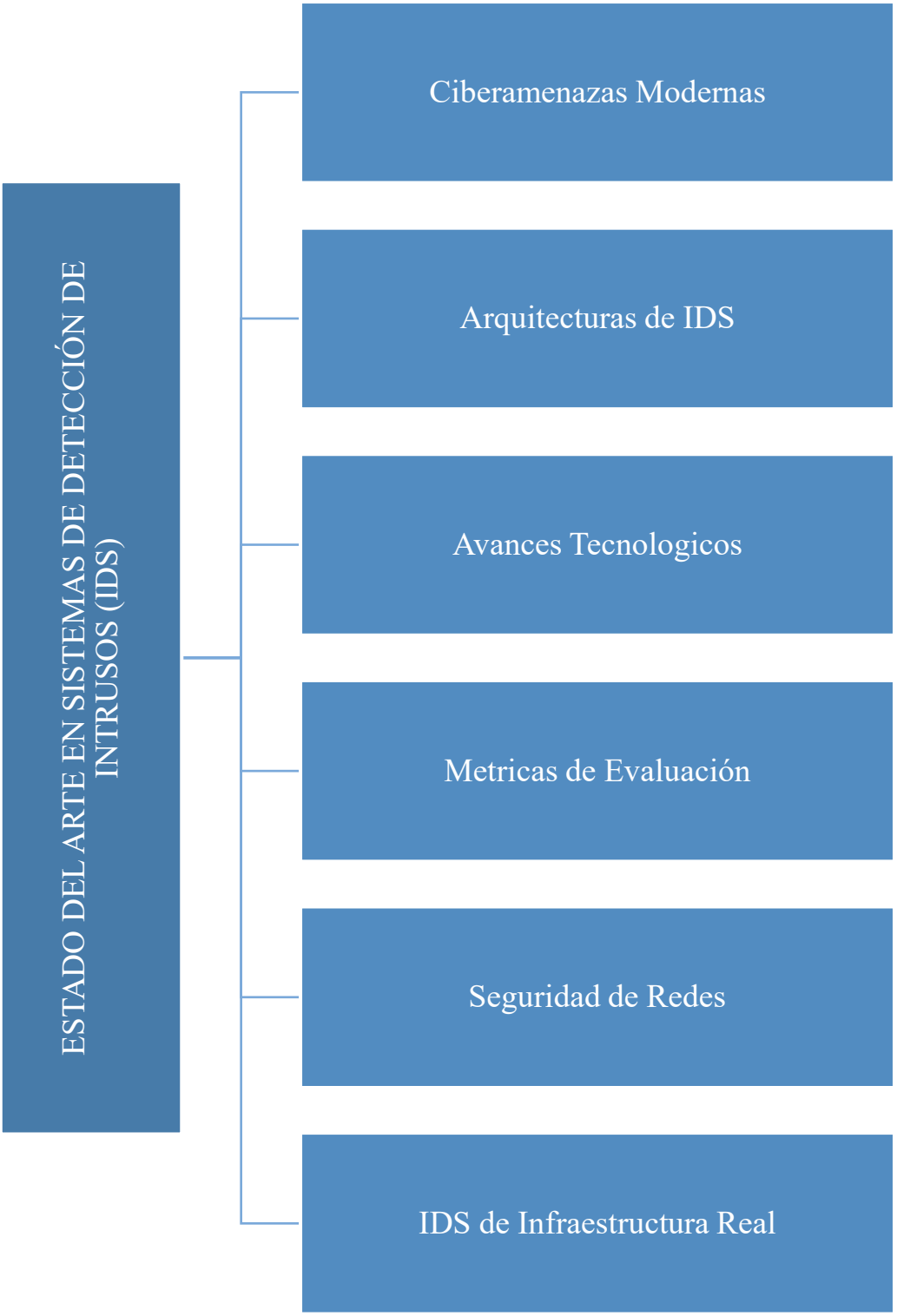
Tabla 4 - Componentes Claves y su Función

COMPONENTES	FUNCIÓN	RELEVANCIA EN EL PROYECTO
Procesador	Ejecuta RouterOS y reglas de detección	Soporta análisis de tráfico en tiempo real
Puertos SFP/Gigabit	Conecta dispositivos de Red	Monitorean tráfico para el NIDS
Firewall	Filtra paquetes Sospechosos	Detecta ataques como DoS o MITM
Sistema de Script	Automatiza alertas y tareas	Envía notificaciones por correo
Memoria	Almacena logs y configuraciones	Registra datos de pruebas

- **Aplicaciones en Seguridad de Redes**

Los equipos MikroTik son ideales para implementar medidas de seguridad como IDS, VPN y firewalls en entornos académicos. En el proyecto, el CRS310 configura un NIDS para detectar amenazas como inyecciones de payload, optimizando la seguridad del laboratorio [22].

2.2.8 Estado del Arte del Proyecto



CAPÍTULO III

3.1 Diseño del Escenario Experimental

El modelo experimental que se plantea tiene como objetivo implementar y evaluar un Sistema de Detección de Intrusos (IDS) de tipo híbrido en el Laboratorio de Telecomunicaciones de la Universidad Estatal Península de Santa Elena (UPSE). Este modelo experimenta la mezcla de una infraestructura física asociada a equipos MikroTik, equipos de fibra óptica y dispositivo WiFi Pineapple HAK5 como se me muestra en la *Figura 13,* con una infraestructura virtualizada donde corren herramientas de especialización para el análisis y vigilancia de la seguridad.

La forma en que se presenta el diseño del escenario implica que el tráfico que se produce en el laboratorio así como legítimo y malicioso que puedan ser capturado, duplicado y enviado a un servidor IDS donde se le pueda inspeccionar en tiempo real. Con ello se asegura que las pruebas de ciberseguridad sean realizadas a cabo en un modelo controlado, es decir, que se pueda garantizar que una red de ciberseguridad que sea ejecutada en red de una manera controlada y donde se avance hacia el cumplimiento de los objetivos específicos del proyecto.

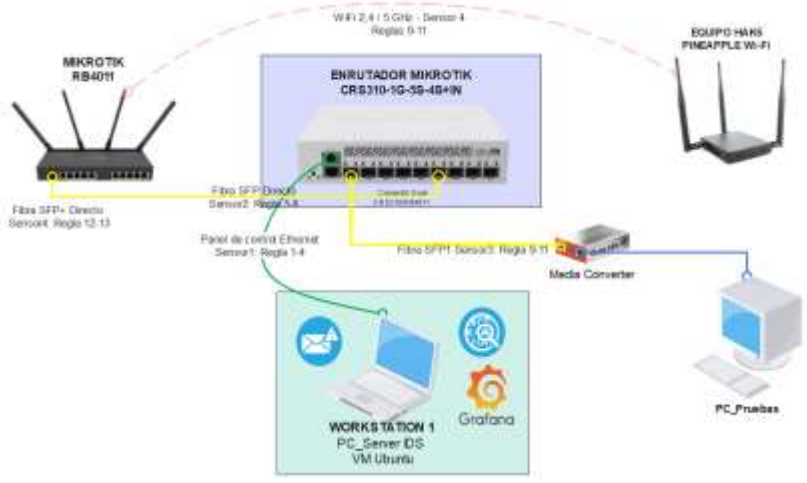


Figura 4 - Diseño Creado para la implementación de la Propuesta

3.1.1 Topología general del sistema IDS-Híbrido

La arquitectura de red se plasma sobre el router MikroTik CRS310 que se convierte en el núcleo de conmutación. En este router de conmutación se conecta el router RB4011, que habilita el acceso a la red cableada e inalámbrica para los usuarios del laboratorio y al mismo tiempo el dispositivo WiFi Pineapple para realizar el ataque de tráfico malicioso sobre la red inalámbrica. Por otra parte se integra una PC/Workstation que determinará el tráfico que recibe conocido como espejado esto es mediante port Mirroring para hacer el análisis de la red y su tráfico mediante las herramientas IDS.

El flujo de trabajo es el siguiente:

- Tráfico legítimo y de prueba, la cual pasa por el equipo MikroTik CRS310 / RB4011.
- El CRS310 es quien replica el tráfico hacia la PC Server IDS.
- Suricata y Zeek analizan el tráfico.
- Promtail envía los registros a Loki.
- Grafana muestra los eventos en paneles de monitorización.

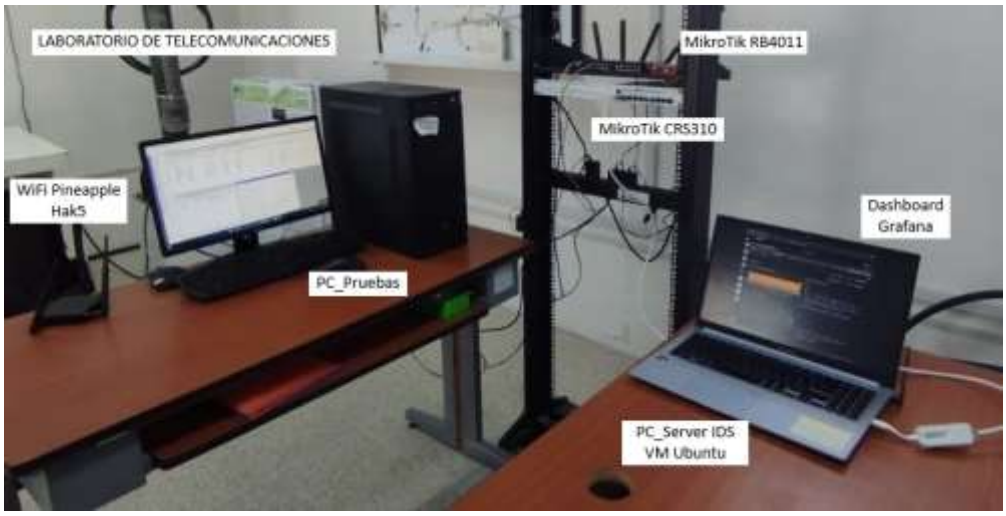


Figura 5 - Entorno de Red Realista en el Laboratorio de Telecomunicaciones / Upse

3.1.2 Diagrama de Flujo del Funcionamiento del IDS

El diagrama de flujo representa el funcionamiento del Sistema de Detección de Intrusos (IDS) implementado en el router MikroTik CRS310 mostrados en la *Ilustración 2*. El proceso inicia con la configuración de puertos y sensores IDS, seguido del monitoreo continuo del tráfico de red. Cuando se detecta una actividad sospechosa, se clasifica el evento, se genera una alerta automática desde RouterOS y se registra en la estación de monitoreo. Finalmente, se verifica la precisión para ajustar reglas o confirmar la estabilidad del sistema.

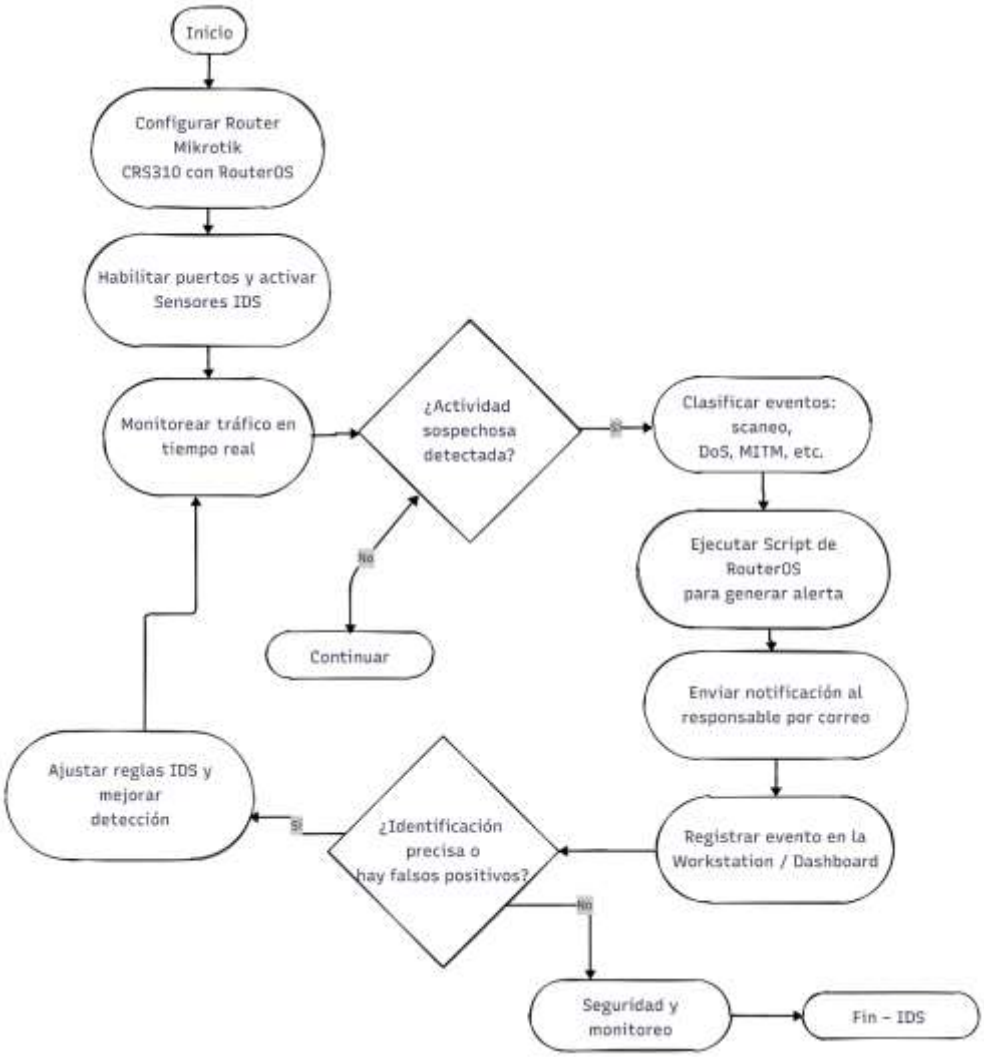


Ilustración 2 – Funcionamiento del IDS en un diagrama de Flujo

3.1.3 Componentes Físicos Utilizados

En la *Tabla 5* se resume los componentes físicos empleados en el escenario experimental.

Tabla 5 - Componentes Físicos del escenario experimental

COMPONENTES	FUNCIÓN PRINCIPAL	UBICACIÓN / ROL DE ESCENARIO
MikroTik CRS310-1G-5S-4S+IN	Núcleo de conmutación y port mirroring	Centro de la topología, enlace con servidor IDS
MikroTik RB4011iGS+5HacQ2HnD	Router de acceso cableado e inalámbrico	Acceso para-PCs del laboratorio y red WiFi de pruebas
PC_Pruebas	Generación de ataques controlados	Punto de ataque sobre la red del laboratorio
PC_Server IDS (Servidor IDS)	Análisis IDS, almacenamiento y visualización	Nodo de monitoreo y procesamiento de logs
Cables Ethernet / SFP+	Interconexión física	Enlaces entre CRS310, RB4011 y servidor IDS

Los componentes físicos permiten recrear un entorno de red realista mostrados en la *Figura 14*, donde el tráfico generado por los usuarios y por el WiFi Pineapple es dirigido hacia el CRS310 y replicado hacia la estación de trabajo para su posterior análisis.

3.1.4 Componentes Virtuales y Software

En el servidor IDS se despliega un conjunto de herramientas de software orientadas a la detección, recolección y visualización de eventos de seguridad:

Suricata para Sensores Integrados

Herramienta IDS/IPS de alto rendimiento basada en reglas. **Su** funcionalidad en el proyecto es analizar el tráfico replicado desde el MikroTik port mirroring y genera alertas en formato (eve. json) y (fast.log) cuando detecta patrones de ataque o tráfico anómalo. Uno de sus requisitos básicos es correr sobre GNU/Linux, interfaz de red en modo promiscuo y recursos de CPU/RAM moderados.

Figura 6 - Suricata

Zeek para Sensores Híbridos

Plataforma de análisis de tráfico orientada a comportamiento. **Su** funcionalidad en el proyecto es genera registros detallados de conexiones (conn.log) y de protocolos (http.log, dns.log, etc.) a partir del mismo tráfico enviado al servidor IDS, complementando las alertas de Suricata con contexto adicional. Uno de sus requisitos básicos es el sistema GNU/Linux y buena capacidad de disco para almacenar logs.

Figura 7 - Zeek

Loki

Sistema de almacenamiento e indexación de logs orientado a series de tiempo. Su funcionalidad en el proyecto es actuar como "repositorio central" donde se guardan los logs enviados por Promtail la cual están Suricata y Zeek, permitiendo consultas posteriores desde Grafana. Como requisitos básicos es ejecutarse como servicio la cual normalmente en contenedor Docker con acceso a disco para la retención de logs.

Figura 8 - Loki Grafana

Promtail

Agente colector de logs. Su funcionalidad en el proyecto es de leer los archivos de logs de Suricata y Zeek en el servidor IDS les agrega etiquetas como el origen el tipo de log su interfaz. Esto los envía a Loki para su almacenamiento, como requisito básico es ser instalado en el mismo servidor donde se generan los logs o con acceso a sus rutas.

Figura 9 - Comportamiento de Promtail

Grafana

Grafana transforma datos crudos en visualizaciones intuitivas, instalado en la Workstation. Su descripción enfatiza la capacidad para Dashboard personalizados. En el proyecto integra logs de RouterOS para monitorear sensores y alertas en tiempo real, ayudando a identificar patrones y ajustar diferentes reglas, lo que hace el análisis más accesible y colaborativo.

Figura 10 - Software Grafana

Docker Engine / Docker Compose

Plataforma de contenedores Docker y herramienta de orquestación declarativa Compose tiene como funcionalidad en el proyecto en facilitar el despliegue de Loki, Promtail y Grafana y si se desea, Suricata/Zeek como contenedores, simplificando instalación, actualización y réplica del entorno IDS. Es requerido para el sistema operativo GNU/Linux compatible con Docker y recursos suficientes de CPU/RAM para los contenedores.

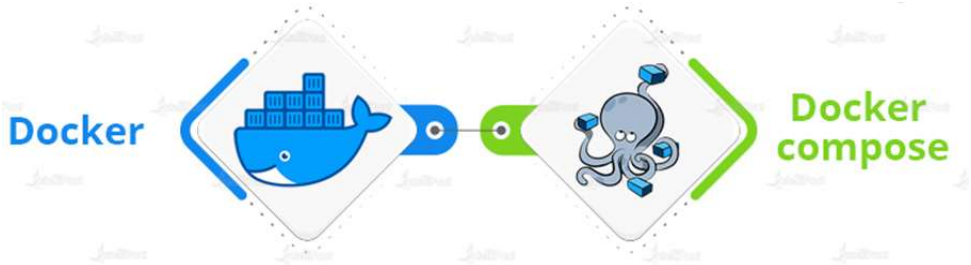


Figura 11 - Plataformas Docker [35]

3.1.5 Requerimiento de Hardware y Software

Para la implementación del escenario se consideran los siguientes requerimientos mínimos:

Router MikroTik CRS310-1G-5S-4S+IN

Este router como se muestra en la *Figura 14* actúa como el núcleo central del sistema con su puerto Gigabit Ethernet y puertos SFP/SFP+ que permiten conexiones versátiles. Su descripción destaca la robustez para manejar tráfico alto, en el proyecto se encarga de los sensores cableados y fibra, facilitando la detección de amenazas como DoS mediante conexiones directas SFP a otros dispositivos. Esta elección fomenta una implementación eficiente alineada con entornos académicos donde la flexibilidad es clave.

Figura 12 – MikroTik CRS310-1G-5S-4S+IN

Router MikroTik RB4011iGS+5HacQ2HnD-IN

Complementando al CRS310, este router mostrado en la *Figura 15* cuenta con antenas WiFi dual-band añade el toque inalámbrico al proyecto. Su descripción resalta los 10 puertos Gigabit y SFP+, ideales para extender la red. En el contexto del IDS, maneja los sensores inalámbricos, capturando amenazas como rogue AP en wlan1/wlan2 y se conecta directamente al CRS310 vía SFP+ para un flujo unificado. Esto enriquece el laboratorio, permitiendo pruebas híbridas sin complicaciones.

Figura 13 - MikroTik RB4011iGS+5HacQ2HnD-IN

Workstation PC

La Workstation equipada en el laboratorio con un sistema operativo Windows que sirve como el cerebro del monitoreo, conectada vía Ethernet a ether1 del CRS310. Su descripción enfatiza su rol en ejecutar Grafana y VM para pruebas. En el proyecto centraliza el análisis de alertas y datos de sensores, simplificando el Setup y permitiendo ajustes iterativos sin hardware adicional lo que hace el IDS más accesible para un entorno académico.

Cables y Conexiones Auxiliares

Los cables de fibra óptica amarillos LC-LC y Ethernet cian RJ45 reutilizados son los lazos que unen el sistema. En el proyecto facilitan conexiones directas SFP/SFP+ entre CRS310 y RB4011 y el Ethernet a la Workstation asegurando un flujo de tráfico estable para los sensores y pruebas.

3.1.6 Topología en Flujo de Trafico

La topología adopta un **modelo híbrido en estrella con elementos en malla**, combinando la centralidad del CRS310 como núcleo principal con conexiones directas y distribuidas. El CRS310 actúa como el centro de la estrella, conectando la Workstation vía ether1, el RB4011 vía

sfp+ con cable de fibra óptica, y sfp para pruebas adicionales, mientras que el RB4011 extiende la red en malla con wlan1/wlan2 hacia el Hak5 Pineapple WiFi.

Esta estructura crea un flujo de tráfico unificado, aislado en VLAN 10 para pruebas seguras, donde el tráfico legítimo e ilegítimo se canaliza desde Hak5 a RB4011, luego a CRS310, y finalmente a la Workstation PC_Server IDS para análisis.

El flujo de tráfico se diseña para optimizar la detección: los ataques simulados del Hak5 activan los sensores, pasando por las reglas IDS en RouterOS antes de generar alertas. Este enfoque híbrido asegura flexibilidad y escalabilidad.

3.2 Configuración de la Infraestructura de Red

La infraestructura de red se configura tomando al MikroTik CRS310 como núcleo del escenario, al RB4011 como router de acceso la cual su comunicación es cableada/inalámbrico y a una PC servidor IDS conectada al puerto espejo. En la *Tabla 6* se resume el esquema de direccionamiento utilizado.

Tabla 6 - Esquema de direccionamiento IP del escenario experimental

DISPOSITIVO	INTERFAZ	RED	DIRECCIÓN IP / MÁSCARA	ROL PRINCIPAL
CRS310-IDS (Router / Switch Tráfico)	**ether1**	WAN (USB Tethering del Teléfono)	**192.168.10.1/24** (DHCP)	Entrada de Internet hacia el laboratorio
CRS310-IDS	**BR bridge-LAB (LAN)**	LAN principal del laboratorio	**192.168.10.1/24**	Puente interno para PCs, VMs e IDS
CRS310-IDS	**sfp1**	LAN Laboratorio (VM)	**192.168.10.0/24** (DHCP desde CRS310)	Conexión PC/VM donde corre Suricata y Zeek
CRS310-IDS	**sfp-sfpplus1**	Enlace hacia RB4011	**Sin IP** (tránsito/mirroring)	Envío de tráfico espejo hacia el sensor IDS

RB4011-IDS (Sensor de Captura IDS)	**ether2**	Gestión	**192.168.88.50/24** (DHCP desde CRS310)	Puerto de administración del equipo IDS
RB4011-IDS	**BR-IDS**	Red de análisis IDS	**192.168.10.2/24**	Segmento lógico para análisis IDS (Suricata/Zeek)
RB4011-IDS	**sfp-sfpplus1**	Tráfico espejo	**Sin IP**	Recepción de tráfico para análisis (entrada del mirroring)
Laptop / VM (Suricata + Zeek)	Adaptador Bridge	LAN IDS	**192.168.10.100/24**	Motor principal de análisis de intrusiones
WiFi Pineapple VII	wlan0	Red de pruebas	**172.16.42.1/24** (default Pineapple)	Generación de ataques WiFi y pruebas de intrusión

El puerto sfp-sfpplus1 del router MikroTik RB4011 opera en modo espejo (SPAN) y por tanto no posee dirección IP asignada. Esto se debe a que un puerto de captura en un IDS pasivo no debe generar tráfico ni participar en la capa 3, ya que su única función es recibir flujos duplicados de la red principal para su análisis mediante Suricata y Zeek.

La ausencia de direccionamiento IP en esta interfaz garantiza que el sensor permanezca pasivo, evitando loops de red y así eliminando tráfico no deseado y asegura la integridad del análisis.

3.2.1 Configuración Base del MikroTik CRS310

En el CRS310 se realizó la configuración base que permite operar como núcleo de la red de laboratorio:

Asignación de Direcciones IP

En el equipo se configuró una IP de gestión en la red del laboratorio 192.168.10.1/24 sobre un bridge que agrupa los puertos conectados a la PC Server IDS y al RB4011, como se muestra en la *Figura 24*.

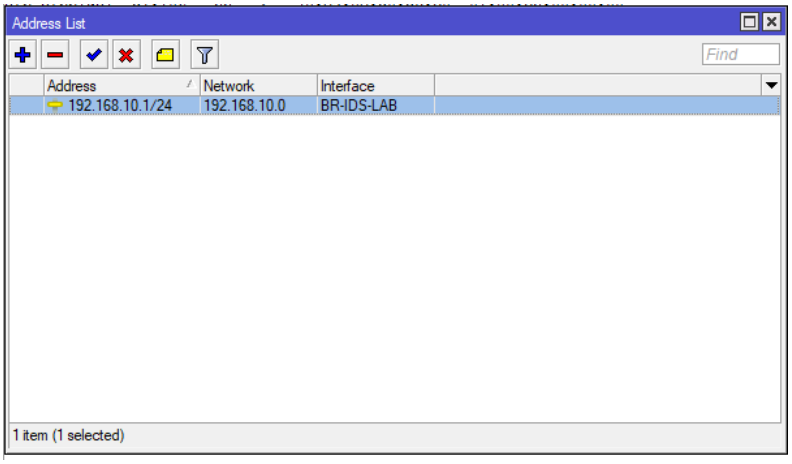


Figura 14 – Address List / MikroTik CRS310

Configuración del DHCP / Gateway

Se habilitó un servidor DHCP en la red 192.168.10.0/24, entregando direcciones a los equipos que se conectan directamente al CRS310 *Figura 25*.

El CRS310 actúa como Gateway local del laboratorio y reenvía el tráfico hacia la red externa a través de ether1.

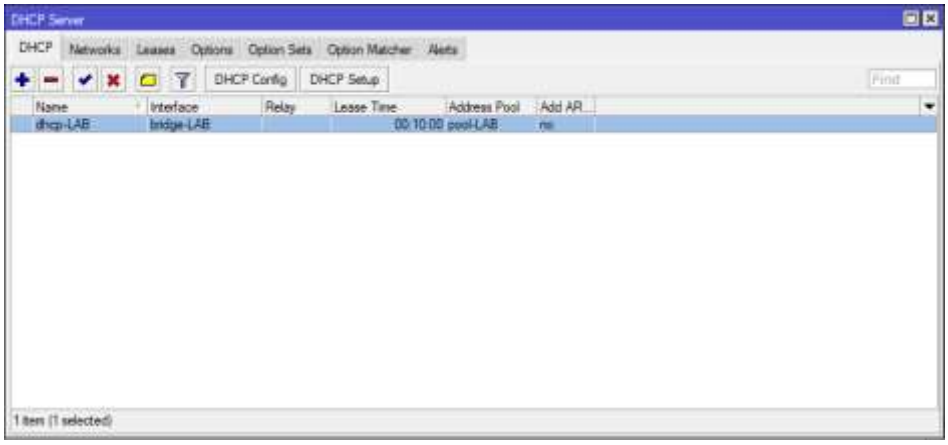


Figura 15 - Habilitación DHCP MikroTik CRS310

Habilitación de Acceso a Internet

- ✓ Se definió una ruta por defecto (default route) apuntando al router de la red institucional.
- ✓ Se configuró NAT para permitir que las máquinas virtuales y el servidor IDS accedan a Internet, por ejemplo para actualizar paquetes o descargar imágenes de contenedores.

Con esta configuración base, el CRS310 concentra todo el tráfico del escenario y está preparado para enviar una copia hacia el servidor IDS.

```
Proyecto de Implementacin de Sensores IDS
Autor: Gregory Ricardo
Laboratorio de Telecomunicaciones - UPSE
[ids-admin@CRS310-IDS] > /ip dhcp-client add interface=ether1 disabled=no
failure: dhcp-client on that interface already exists
[ids-admin@CRS310-IDS] > /interface bridge add name=bridge-LAB
[ids-admin@CRS310-IDS] > /interface bridge port add bridge=bridge-LAB interface=sfp1
failure: device already added as bridge port
[ids-admin@CRS310-IDS] > /interface bridge port add bridge=bridge-LAB interface=sfp2
[ids-admin@CRS310-IDS] > /interface bridge port add bridge=bridge-LAB interface=ether2
invalid value for argument interface:
    input does not match any value of interface
    input does not match any value of interface-list
[ids-admin@CRS310-IDS] > /interface bridge port add bridge=bridge-LAB interface=sfp1
failure: device already added as bridge port
[ids-admin@CRS310-IDS] > /interface bridge port add bridge=bridge-LAB interface=sfp2
failure: device already added as bridge port
[ids-admin@CRS310-IDS] > /interface bridge port add bridge=bridge-LAB interface=ether2
invalid value for argument interface:
    input does not match any value of interface
    input does not match any value of interface-list
[ids-admin@CRS310-IDS] > /ip address add address=192.168.10.1/24 interface=bridge-LAB
[ids-admin@CRS310-IDS] > /ip pool add name=pool-LAB ranges=192.168.10.10-192.168.10.200
[ids-admin@CRS310-IDS] > /ip dhcp-server add name=dhcp-LAB interface=bridge-LAB address-pool=pool-LAB disabled=no
[ids-admin@CRS310-IDS] > /ip dhcp-server network add address=192.168.10.0/24 gateway=192.168.10.1 dns-server=8.8.8
.8
[ids-admin@CRS310-IDS] > /ip firewall nat add chain=srcnat out-interface=ether1 action=masquerade
[ids-admin@CRS310-IDS] >
```

Figura 16 - Terminal de Winbox CRS310 - Configuración Manual por código

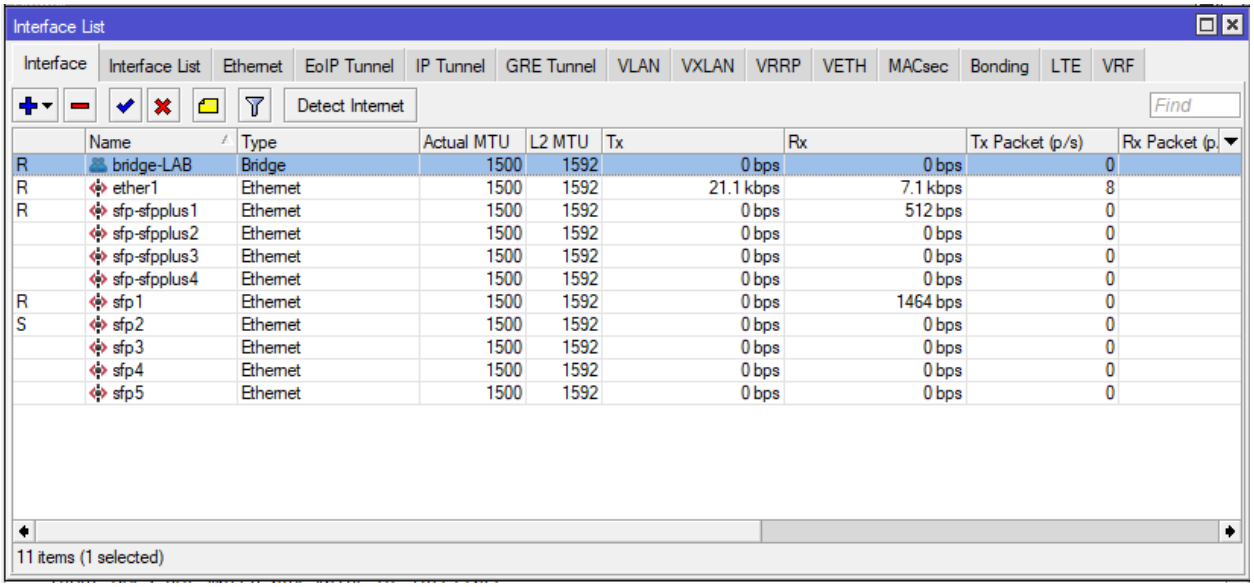


Figura 17 - Lista de Interfaces Habilitadas

3.2.2 Integración del MikroTik RB4011

La incorporación del router MikroTik RB4011iGS+5HacQ2HnD-IN dentro de la arquitectura IDS diseñada para el laboratorio permite disponer de un segundo punto de distribución de red y un nodo adicional para la generación y análisis de tráfico *Figura 28*. Su función principal consiste en operar como un router de acceso dentro del segmento 192.168.20.0/24, facilitando tanto el tránsito de paquetes como la obtención de métricas utilizadas posteriormente en los sensores IDS basados en Suricata y Zeek.

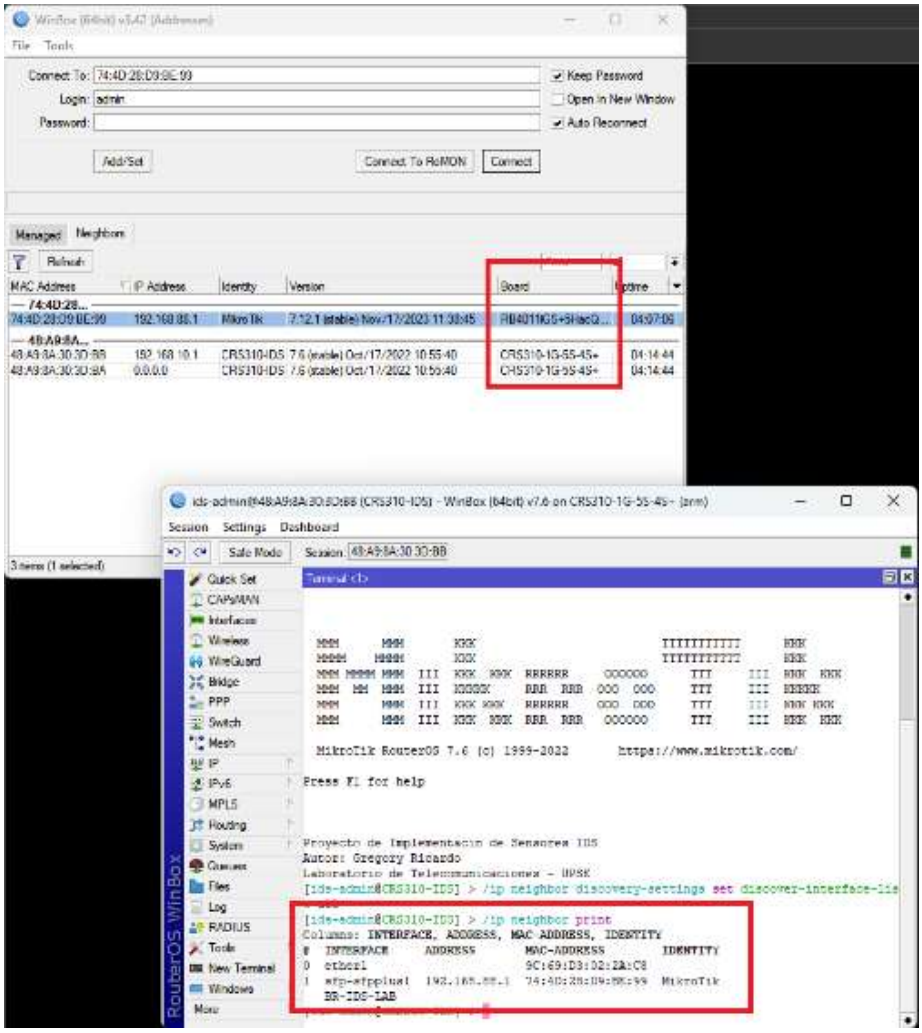


Figura 18 – Integración del RB4011 y CRS310

Configuración de Subredes

✓ Para establecer la nueva subred de labores de monitoreo interno, se configuró la interfaz de bridge del RB4011 mediante los siguientes parámetros:

```
[admin@RB4011-IDS] > /ip address add address=192.168.20.1/24 interface=BR-LAB
[admin@RB4011-IDS] > /ip address print
Columns: ADDRESS, NETWORK, INTERFACE
# ADDRESS            NETWORK        INTERFACE
;;; defconf
0 192.168.88.1/24    192.168.88.0   bridge
1 192.168.20.1/24    192.168.20.0   BR-LAB
```

Figura 19 – Puerta de Enlace y Bridge

Esta dirección mostrada en la *Figura 29* opera como puerta de enlace predeterminada del segmento 192.168.20.0/24, proporcionando un dominio de broadcast independiente para pruebas de tráfico, simulación de amenazas y análisis con IDS.

Pool de Direcciones y DHCP Server

Con el fin de automatizar la asignación de direcciones IP para los equipos conectados al RB4011, se habilitó un servidor DHCP interno:

```
[admin@RB4011-IDS] > /ip pool add name=pool-LAB ranges=192.168.20.10-192.168.20.200

[admin@RB4011-IDS] > /ip dhcp-server add name=dhcp-LAB interface=BR-LAB address-poo
l=pool-LAB disabled=no
[admin@RB4011-IDS] > /ip dhcp-server network add address=192.168.20.0/24 gateway=19
2.168.20.1 dns-server=8.8.8.8
[admin@RB4011-IDS] > /ip dhcp-server print
Columns: NAME, INTERFACE, ADDRESS-POOL, LEASE-TIME
# NAME      INTERFACE  ADDRESS-POOL  LEASE-TIME
0 defconf   bridge     default-dhcp  10m
1 dhcp-LAB  BR-LAB     pool-LAB      30m
```

Figura 20 - Asignación dinámica controlada para IDS

Esta configuración que se muestra en la *Figura 30* asegura una asignación dinámica ordenada, permitiendo realizar pruebas controladas de tráfico IDS desde máquinas conectadas al RB4011.

Configuración de Rutas Estáticas

- ✓ El RB4011 utiliza como default route la IP del CRS310 (192.168.10.1), de manera que todo el tráfico de la red 192.168.20.0/24 salga hacia Internet a través del CRS310.
- ✓ En el CRS310 se añadió una ruta estática hacia la red 192.168.20.0/24 con siguiente salto 192.168.10.2, garantizando el retorno del tráfico.

Implementación de Neighbor Discovery

Para facilitar la detección automática del RB4011 desde otros equipos MikroTik, se habilitó la lista de interfaces de descubrimiento:

```
[admin@RB4011-IDS] > /interface list add name=discover
[admin@RB4011-IDS] > /interface list member add list=discover interface=BR-LAB
[admin@RB4011-IDS] > /ip neighbor discovery-settings set discover-interface-list=di
scover
```

Figura 21 - Detección entre equipos RB4011 y CRS310

Esto permite que el router aparezca en herramientas como Winbox, reforzando la visibilidad dentro de la infraestructura administrada como se muestra en la *Figura 31*.

3.2.3 Implementación del Port Mirroring para Sensores IDS

Con el propósito de capturar tráfico en tiempo real y replicarlo hacia los sensores IDS ubicados en el servidor, se configuró la funcionalidad de port mirroring en el CRS310. El SFP+ utilizado para la conexión hacia el RB4011 se declaró fuente de monitoreo y el puerto Ethernet del servidor IDS se estableció como destino:

```
[ids-admin@CRS310-IDS] > /interface ethernet switch set switch1 mirror-source=sfp-s
fpplus1 mirror-target=ether1
[ids-admin@CRS310-IDS] > /interface ethernet switch print detail
Flags: I - invalid
 0   name="switch1" type=Marvell-98DX226S mirror-source=sfp-sfpplus1
     mirror-target=ether1 mirror-egress-target=none l3-hw-offloading=no
```

Figura 22 - Ejecución de port mirroring entre CRS310 y RB4011

En la *Figura 32* se muestra la configuración y nos indica que fue correctamente implementada y verificada en el CRS310. La función permanece activa y operativa para pruebas posteriores o para conexión mediante un patchcord compatible.

Verificación con Sniffer y pruebas de tráfico

Se utilizó la herramienta de sniffer del Mikrotik y utilidades como tcpdump en el servidor IDS para confirmar que:

- ✓ El tráfico originado en la red 192.168.10.1/24 y en los hosts del laboratorio llega efectivamente al puerto espejo.
- ✓ Se observan paquetes tanto legítimos como generados por el WiFi Pineapple durante las pruebas.

```
 0   name="switch1" type=Marvell-98DX226S mirror-source=sfp-sfpplus1
     mirror-target=ether1 mirror-egress-target=none l3-hw-offloading=no
[ids-admin@CRS310-IDS] > /ping 192.168.20.1 interface=sfp-sfpplus1 count=20
  SEQ HOST                                     SIZE TTL TIME       STATUS
    0 192.168.20.1                                                 timeout
    1 192.168.20.1                                                 timeout
    2 192.168.20.1                                                 timeout
    3 192.168.10.1                               84  64 80ms794us  host unreach...
    4 192.168.20.1                                                 timeout
    5 192.168.20.1                                                 timeout
    6 192.168.20.1                                                 timeout
    7 192.168.10.1                               84  64 149ms238us host unreach...
    8 192.168.20.1                                                 timeout
    9 192.168.20.1                                                 timeout
   10 192.168.20.1                                                 timeout
   11 192.168.10.1                               84  64 138ms962us host unreach...
   12 192.168.20.1                                                 timeout
   13 192.168.20.1                                                 timeout
   14 192.168.20.1                                                 timeout
   15 192.168.10.1                               84  64 128ms975us host unreach...
   16 192.168.20.1                                                 timeout
   17 192.168.20.1                                                 timeout
   18 192.168.20.1                                                 timeout
   19 192.168.10.1                               84  64 119ms155us host unreach...
    sent=20 received=0 packet-loss=100%
```

Figura 23 – Verificación y pruebas de trafico

```
Terminal <1>
    9 8.8.8.8                                                      timeout
    sent=10 received=0 packet-loss=100%

[ids-admin@CRS310-IDS] /ip/address> /tool sniffer quick interface=sfp1
Columns: INTERFACE, TIME, NUM, DIR, SRC-MAC, DST-MAC, PROTOCOL, SIZE, CPU
INTE  TIME    N  DIR  SRC-MAC            DST-MAC            PROTO  SI  C
sfp1  0.98    1  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
sfp1  2.982   2  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
sfp1  4.985   3  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
sfp1  6.987   4  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
sfp1  8.989   5  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
sfp1  10.991  6  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
sfp1  12.993  7  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
sfp1  14.996  8  ->   48:A9:8A:30:3D:BB  01:80:C2:00:00:00  802.2  53  1
-- [Q quit|D dump|C-z pause]
```

Figura 24 - Herramienta de verificación Sniffer

Con esta configuración, el servidor IDS recibe una copia transparente del tráfico del laboratorio sin interferir con la comunicación original, permitiendo a Suricata y Zeek analizar los paquetes y generar las alertas que posteriormente se visualizarán en Grafana.

3.3 Implementación de los Sensores IDS

En el servidor IDS se implementaron dos sensores principales que son Suricata y Zeek. Ambos reciben el tráfico que llega desde el port mirroring configurado en el MikroTik CRS310 y trabajan de forma complementaria:

- Suricata se enfoca en la detección basada en reglas (alertas IDS).
- Zeek se orienta al registro detallado de conexiones y protocolos.

3.3.1 Instalación y configuración de Suricata

Suricata se instaló sobre el sistema GNU/Linux del servidor IDS y se configuró para escuchar en la interfaz conectada al puerto espejo del CRS310.

Pasos principales realizados:

1. **Instalación y activación del servicio**

   ✓ Instalación de Suricata desde los repositorios oficiales.

   Se hizo la instalación en la VM Ubuntu con los comandos:

   - *sudo apt update*
   - *sudo apt install -y suricata*

   Al ejecutar *suricata - - version* se verifica la versión instalada como se muestra en la *Figura 35:*

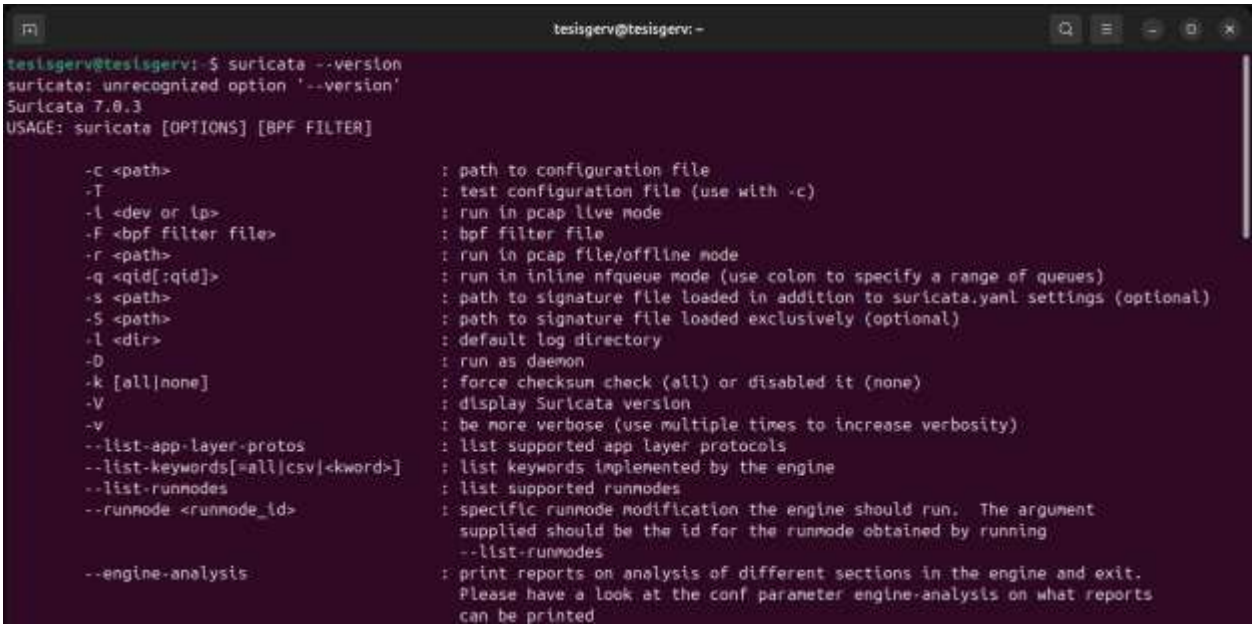


Figura 25 - Versión de Suricata Instalada

Así mismo revisamos si está Activo ejecutando el comando *sudo systemctl status suricata --no-pager* como se muestra en la *Figura 36:*

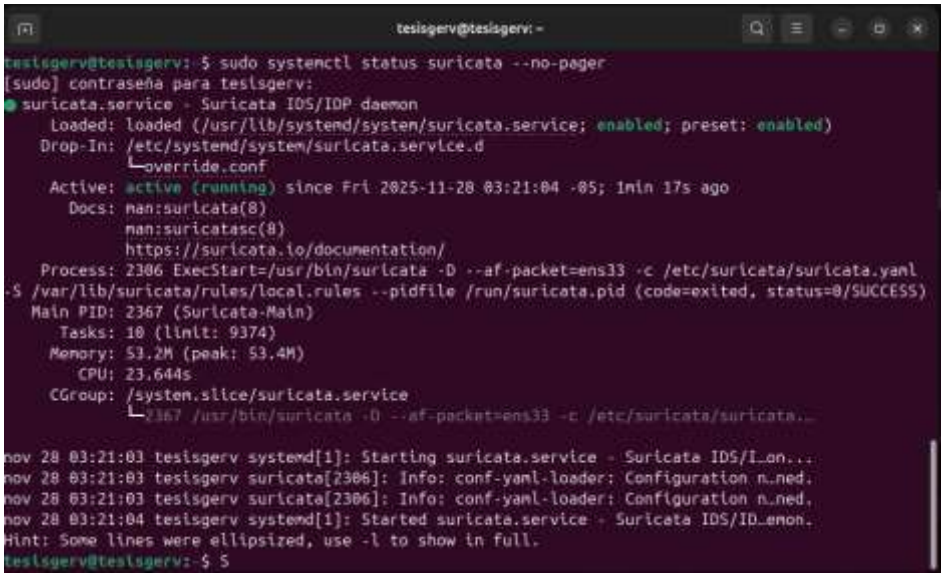


Figura 26 - Activación de Suricata

- ✓ Configuración de la interfaz de captura en el archivo suricata.yaml.

En la *Figura 37* se muestra en que interfaz está usando Suricata al ejecutar *ip a:*

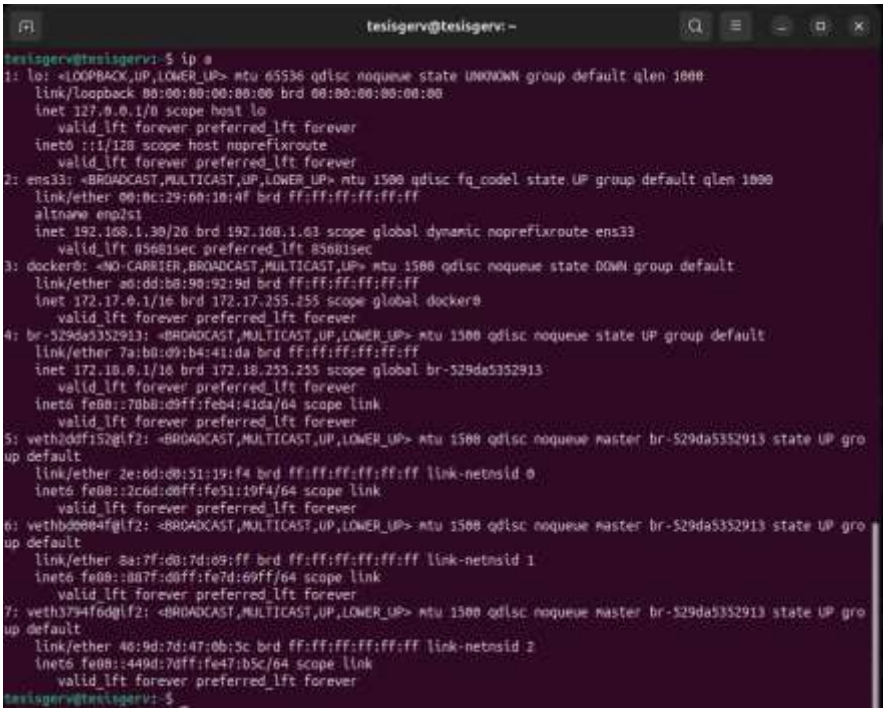


Figura 27 - Interfaz de Suricata

Confirmamos el contenido actual de Suricata.yaml al ejecutar *sudo sed -n '1,200p' /etc/suricata/suricata.yaml* como se muestra en la *Figura 27.* Suricata fue configurado con los grupos de direcciones por defecto, en donde la variable HOME_NET incluye los rangos privados 192.168.0.0/16, 10.0.0.0/8 y 172.16.0.0/12. Esta definición permite al IDS identificar correctamente qué hosts pertenecen a la red local.

La variable *EXTERNAL_NET* fue definida como *!$HOME_NET*, lo que indica que cualquier dirección fuera de los rangos internos será considerada tráfico externo.

Esta clasificación es fundamental para la activación de reglas basadas en direcciones internas vs externas, especialmente para detección de ataques provenientes de Internet o desde dispositivos conectados al laboratorio.

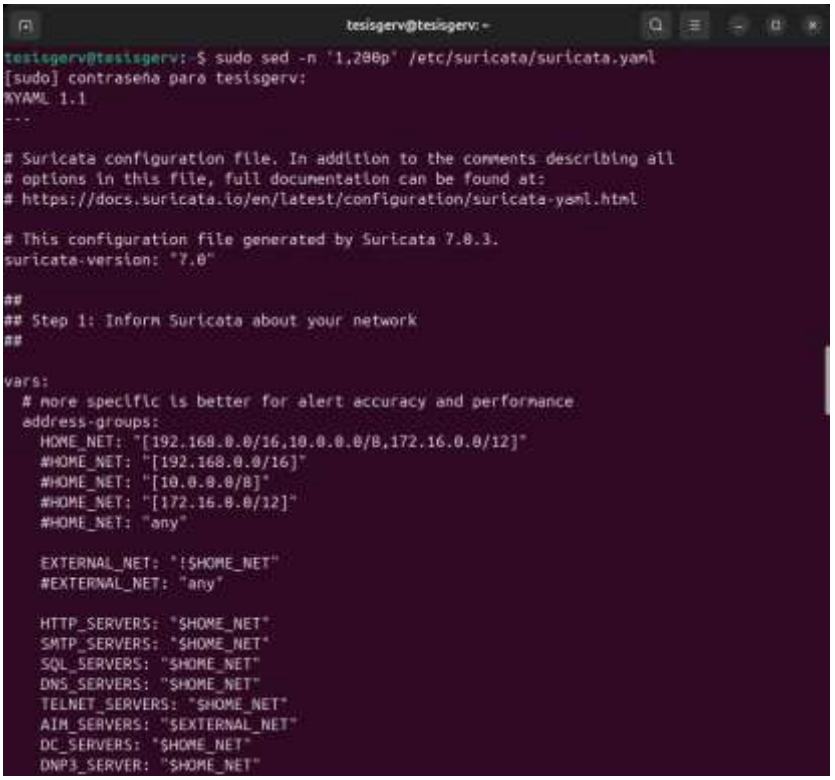


Figura 28 - Contenido Actual de Suricata.yaml

✓ Configuración Reglas Locales en Suricata (local.rules)

Dentro del proyecto, se crearon reglas personalizadas en el archivo:

- */var/lib/suricata/rules/local.rules*

Estas firmas permiten validar el correcto funcionamiento del sensor IDS y detectar patrones específicos utilizados durante las pruebas en el laboratorio de Telecomunicaciones.

La *Figura 39* muestra la configuración de la firma dentro del archivo *local.rules*, utilizada para validar el tráfico HTTP generado en el laboratorio y confirmar la capacidad del IDS para identificar cabeceras manipuladas por el atacante. Visualización de regla local en Suricata correspondiente a la detección del User-Agent LAB-UA-TEST. Al ejecutar *sudo sed -n '1,200p' /var/lib/suricata/rules/local.rules* podemos observar las reglas:

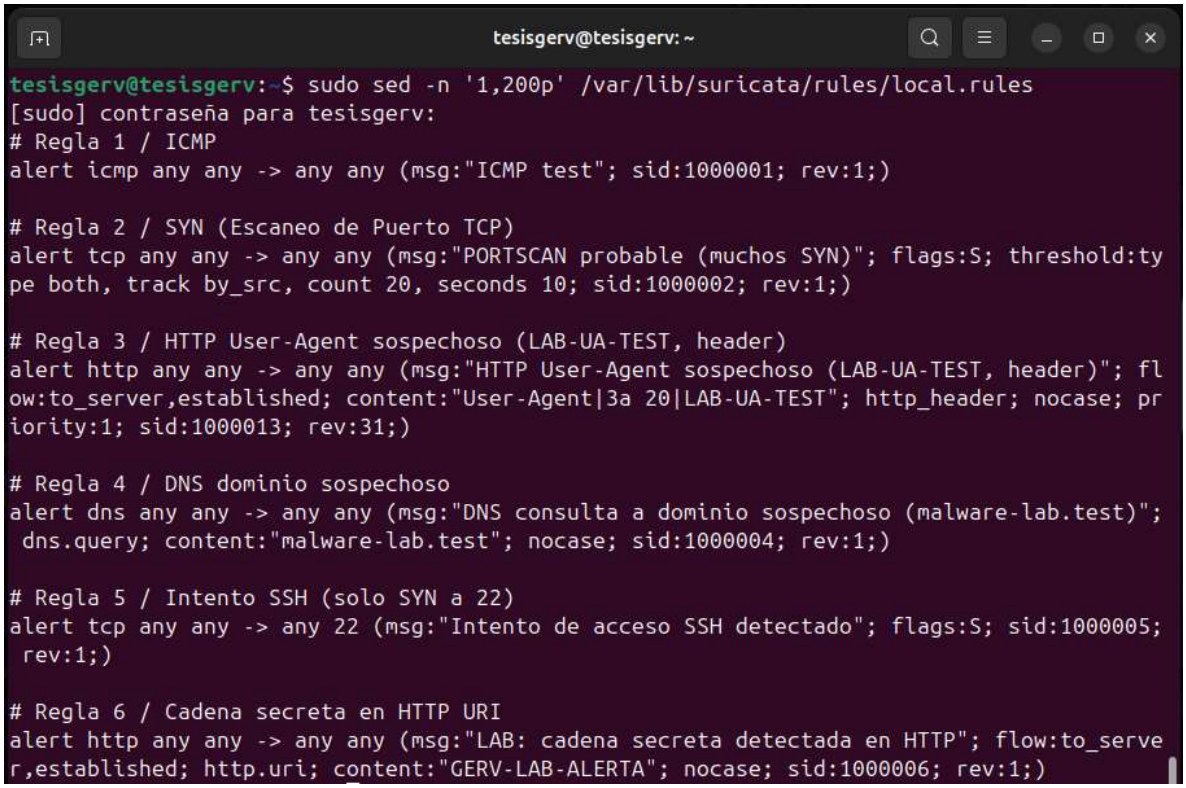


Figura 29- Regla Local en Suricata

Las reglas personalizadas fueron diseñadas para validar el correcto funcionamiento de Suricata y para ellos en la *Tabla 7* tenemos la organización y funcionalidad en le proyecto:

Tabla 7 – Reglas Locales personalizadas del IDS Suricata

N.º	Regla / SID	Comando (Regla Suricata)	Funcionalidad
1	ICMP TestSID: 1000001	`alert icmp any any -> any any (msg:"ICMP test"; sid:1000001; rev:1;)`	Verifica que Suricata captura correctamente tráfico ICMP. Permite validar conectividad y funcionamiento básico del sensor IDS.
2	Detección de PortScan SYNSID: 1000002	`alert tcp any any -> any any (msg:"PORTSCAN probable (muchos SYN)"; flags:S; threshold:type both, track by_src, count 20, seconds 10; sid:1000002; rev:1;)`	Detecta múltiples paquetes SYN en un intervalo corto, simulando un escaneo de puertos (reconocimiento de atacante).
3	HTTP User-Agent Sospechoso (LAB-UA-TEST)SID: 1000013	`alert http any any -> any any (msg:"HTTP User-Agent sospechoso (LAB-UA-TEST, header)"; flow:to_server,established; content:"User-Agent`	Esto comprueba la capacidad del IDS para inspeccionar protocolos de capa 7

4	DNS Dominio SospechosoSID: 1000004	`alert dns any any -> any any (msg:"DNS consulta a dominio sospechoso (malware-lab.test)"; dns.query; content:"malware-lab.test"; nocase; sid:1000004; rev:1;)`	Identifica solicitudes DNS hacia un dominio ficticio utilizado para pruebas de laboratorio. Verifica el análisis de tráfico DNS en Suricata.
5	Intento de Conexión SSHSID: 1000005	`alert tcp any any -> any 22 (msg:"Intento de acceso SSH detectado"; flags:S; sid:1000005; rev:1;)`	Detecta intentos de conexión al puerto 22/TCP (SSH). Útil para simular ataques de fuerza bruta o accesos no autorizados.
6	Cadena Secreta en URL HTTPSID: 1000006	`alert http any any -> any any (msg:"LAB: cadena secreta detectada en HTTP"; flow:to_server,established; http.uri; content:"GERV-LAB-ALERTA"; nocase; sid:1000006; rev:1;)`	Valida la capacidad del IDS para inspeccionar rutas HTTP. Detecta patrones personalizados enviados como simulación de un indicador de compromiso (IOC).

3.4 Verificación y Análisis de Logs Generados por Suricata

En esta sección se describen los comandos utilizados para visualizar los registros generados por Suricata, así como la interpretación de los resultados obtenidos en los archivos fast.log y eve.json. Suricata está generando logs completos y consistentes, la estructura de archivos generados confirma su funcionamiento continuo. El archivo fast.log confirma que las reglas se activan correctamente, el archivo eve.json evidencia datos profundos del tráfico capturado, incluyendo alertas HTTP, ICMP, DNS, etc. La detección del User-Agent LAB-UA-TEST fue exitosa y queda reflejada en JSON.

- ✓ Listar los archivos de Logs:

Al ejecutar ls -lh /var/log/suricata/ muestra la lista de todos los archivos que Suricata genera *Figura 40*. Muestra tamaños, fechas de creación/modificación y tipo de archivo. Este comando permite verificar que Suricata está corriendo correctamente y está generando logs sin errores.

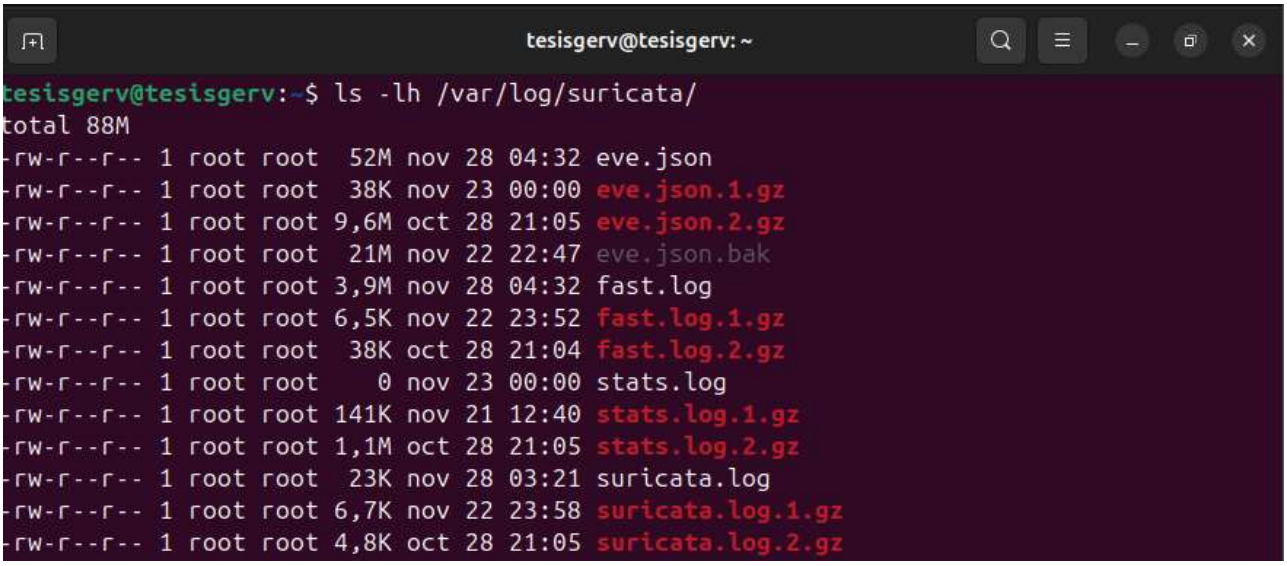


Figura 30 - Archivos de Logs en Suricata

Cuenta con los archivos importantes:

- eve. json = log detallado en formato JSON
- fast.log = log resumido de alertas
- stats.log = estadísticas del motor
- suricata.log = mensajes internos del IDS

✓ Visualizar el fast.log / alertas rápidas

Al ejecutar *sudo tail -f /var/log/suricata/fast.log*. Muestra en tiempo real todas las alertas que Suricata va generando. Se usa para verificar si las reglas personalizadas están siendo activadas correctamente. Indica tráfico ICMP IPv6 constante dentro de la red. En la *Figura 41* demuestra que el sensor está escuchando la interfaz ens33 y las reglas personalizadas se ejecutan correctamente:

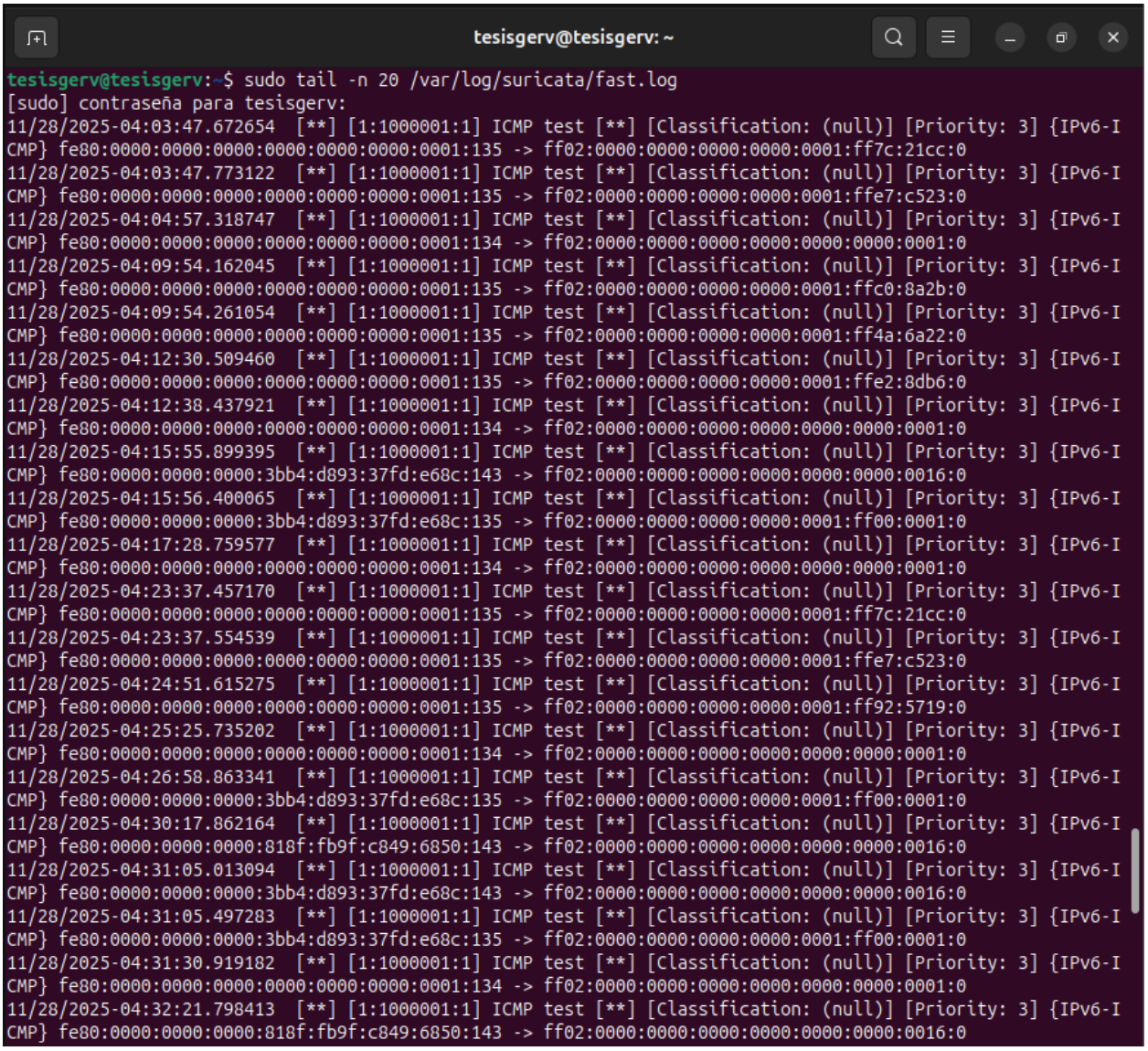


Figura 31 - Ultimas 20 líneas del fast.log ejecutadas

- ✓ Visualizar alertas en eve.json / Modo detallado

Al ejecutar *sudo tail -f /var/log/suricata/eve.json* muestra el archivo más completo de Suricata. Contiene alertas enriquecidas en formato JSON.

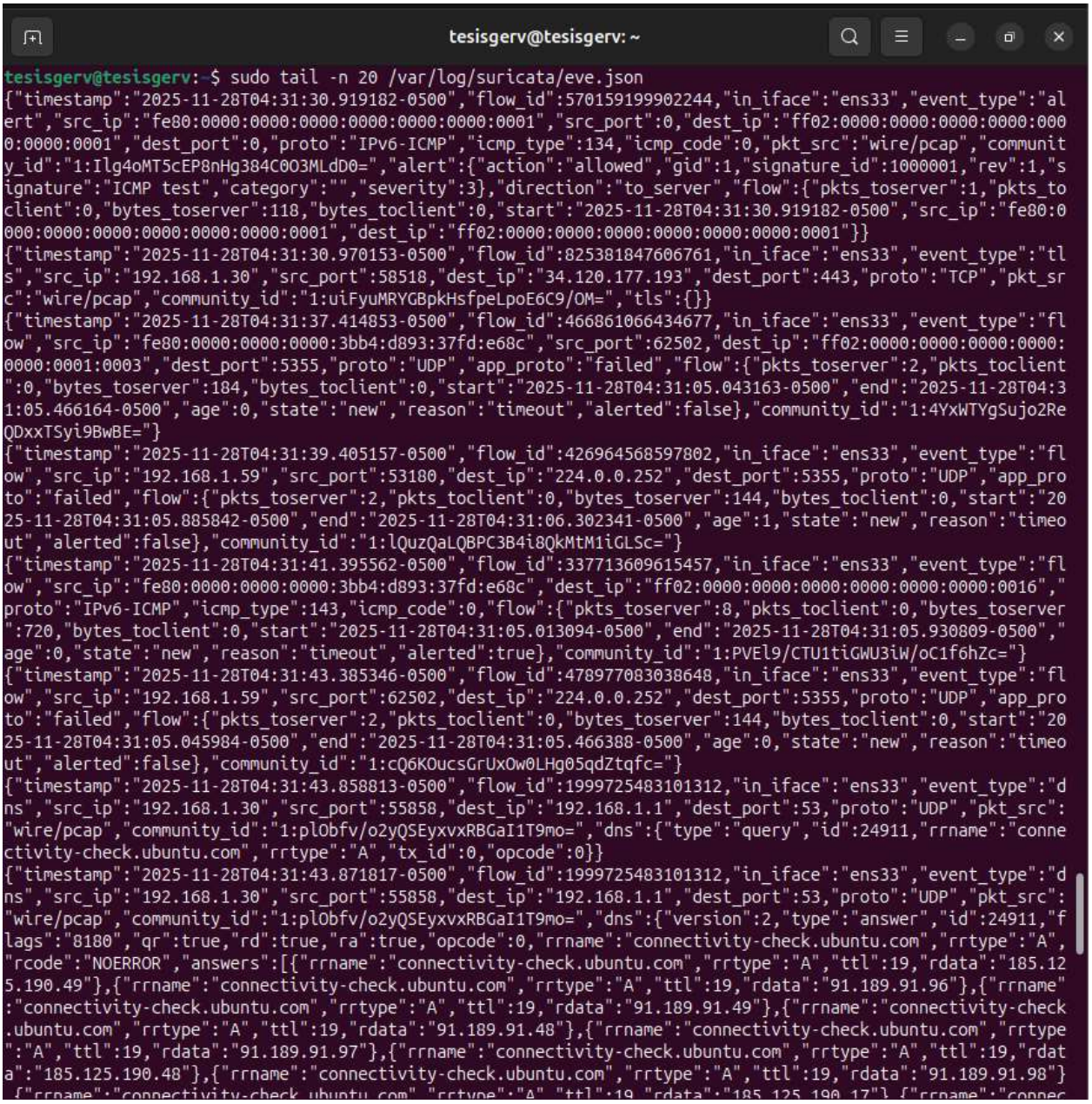


Figura 32 – Ultimas 20 líneas de eve.json ejecutadas

En la *Figura 42* se observa eventos de tipo alert, flow, dns, http, etc.

Datos completos por cada evento:

Tabla 8 – Descripción de cada evento mostrado en los resultados de las Ultimas 20 lineas de eve.json

CAMPO	DESCRIPCIÓN	FUNCIÓN DENTRO DEL ANÁLISIS IDS
timestamp	Fecha y hora exacta en la que ocurrió el evento (formato ISO 8601).	Permite correlacionar eventos, reconstruir línea temporal de un ataque y sincronizar con otros sistemas de monitoreo como Grafana o MikroTik.
in_iface	Nombre de la interfaz de red por donde Suricata capturó el paquete (ens33).	Identifica por qué sensor o NIC entró el tráfico; útil en redes con múltiples interfaces o sensores simultáneos.
event_type	Tipo de evento generado (alert, flow, dns, http, etc.).	Permite clasificar el tipo de tráfico o alerta; sirve para filtrar información según el tipo de análisis requerido.
src_ip / dest_ip	Dirección IP de origen y destino del paquete capturado.	Muestra quién origina y quién recibe la comunicación; fundamental para detectar atacantes, víctimas y relaciones entre activos .
src_port / dest_port	Puerto de origen y puerto de destino del tráfico.	Permite identificar servicios utilizados (80 HTTP, 22 SSH) y posibles intentos de explotación o escaneo.
proto	Protocolo utilizado en la comunicación (TCP, UDP, ICMP, etc.).	Ayuda a entender la naturaleza del tráfico; esencial para validar comportamientos sospechosos por protocolo.
http_user_agent	Cadena informativa enviada por el cliente en tráfico HTTP. En tu proyecto, se activa al detectar el valor modificado "LAB-UA-TEST".	Indica el software/cliente utilizado para hacer una solicitud HTTP. En este proyecto permite detectar modificaciones sospechosas y pruebas de intrusión desde WiFi Pineapple o clientes manipulados.
signature_id	Identificador único de la regla de Suricata que generó la alerta (1000013).	Identifica qué regla se activó. Permite rastrear reglas personalizadas, medir su eficacia y diferenciar entre alertas propias del laboratorio y reglas estándar.

3.5 Validación de Alertas Generadas (User-Agent LAB-UA-TEST)

- ✓ Alertas generadas por la regla personalizada (SID 1000013)

Para comprobar que el sensor IDS basado en Suricata detecta correctamente las solicitudes HTTP con el User-Agent modificado utilizado en las pruebas del laboratorio (LAB-UA-TEST), se ejecutó un conjunto de comandos orientados a filtrar, localizar y mostrar únicamente las alertas originadas por esta regla personalizada.

El primer paso consiste en verificar que el archivo principal de eventos de Suricata (eve.json) contiene alertas generadas por la regla:

- *sudo grep -a '1000013' /var/log/suricata/eve.json*

Al ejecutar este comando busca dentro de eve. json cualquier línea que contenga el valor 1000013. Este número corresponde al signature_id asignado a la regla creada para detectar el User-Agent LAB-UA-TEST. Si el comando muestra líneas con "signature_id": 1000013, significa que Suricata detectó y generó alertas usando la regla definida.

- ✓ Visualización estructurada de alertas usando jq

Una vez confirmada la existencia de alertas, se usa jq para mostrarlas con formato JSON legible, ejecutamos el siguiente comando:

- *sudo jq 'select(.alert.signature_id==1000013)' /var/log/suricata/eve.json*

El comportamiento de *jq* analiza el archivo JSON completo asi mismo con *select(.alert.signature_id==1000013)* extrae únicamente los eventos correspondientes. Muestra con identificación y estructura organizada como se visualiza en la *Figura 40*.

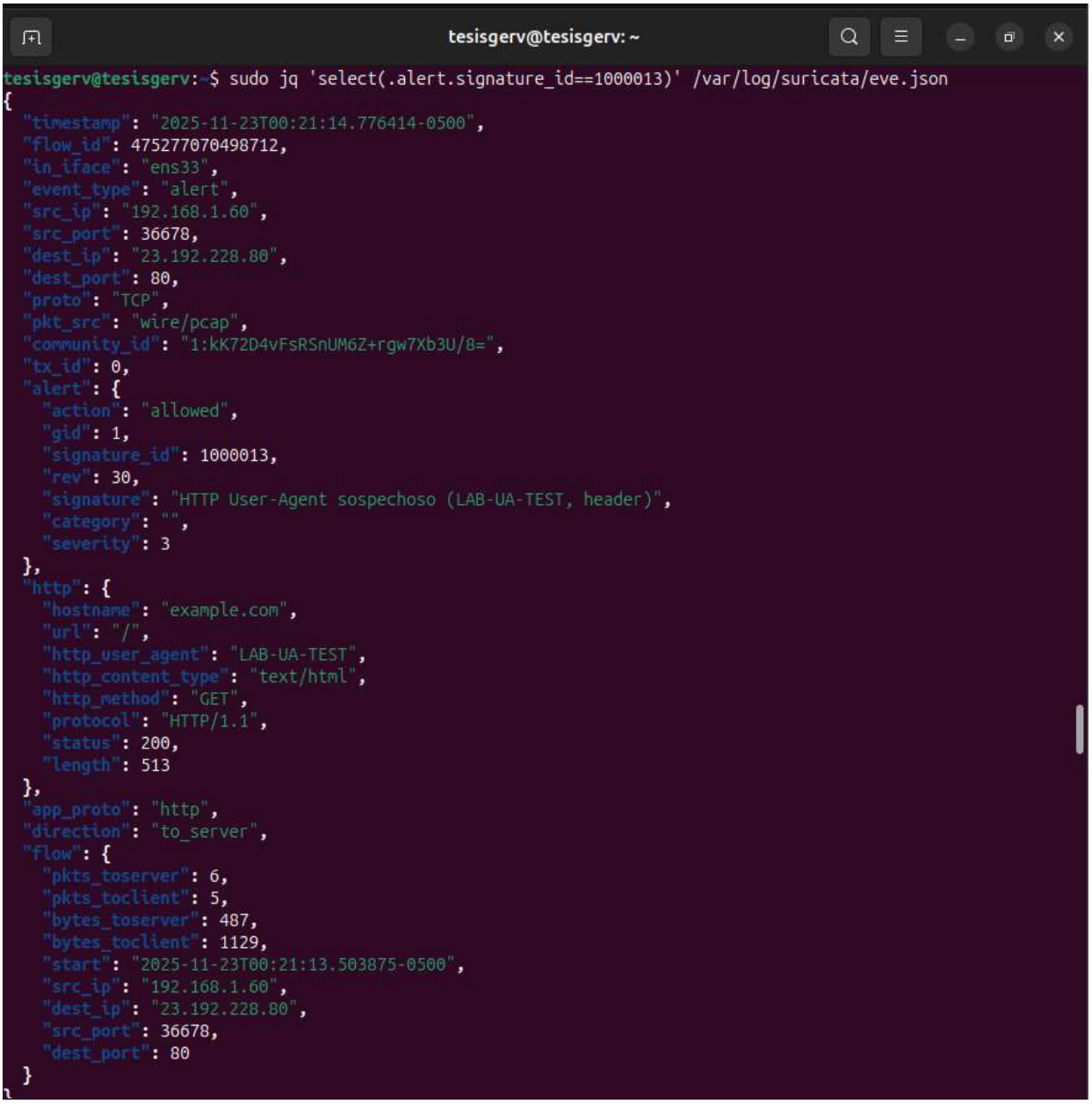


Figura 33 – Visualización de User-Agent LAB-UA-TEST

El resultado permite ver los elementos claves que se analiza en esta sección como en la *Tabla 8* mostrada en la otra sección tales como timestamp exacto de la alerta, src_ip / dest_ip, puertos implicados, User-Agent capturado, Asignature: el mensaje de la regla, severity, flow: número de paquetes y bytes transferidos.

✓ Evidencia de Activación de la Regla LAB-UA-TEST

En la salida mostrada se puede identificar la activación correcta de la regla, tales como se muestran en las siguientes figuras:

Figura 34 - Identificar la Activación correcta de las Reglas

La *Figura 44* demuestra que Suricata inspeccionó el tráfico HTTP, se detectó el encabezado User-Agent modificado. Se activó correctamente la regla creada para pruebas y se generó alerta tanto en fast.log como en eve. json en formato JSON enriquecido.

3.3.2 Instalación y configuración de Zeek

Zeek se utilizó como segundo sensor IDS dentro del entorno del laboratorio esto complementando a Suricata mediante la inspección profunda del tráfico y la generación de logs por protocolo y así se integró un script propio para detección personalizada similar a la regla LAB-UA-TEST usada en Suricata.

1. **Configuración Inicial del servicio Zeek**

Zeek fue instalado desde el repositorio oficial de OpenSUSE Security, siguiendo buenas prácticas para entornos IDS. Una vez instalado, se verificó el estado del servicio:

sudo zeekctl status

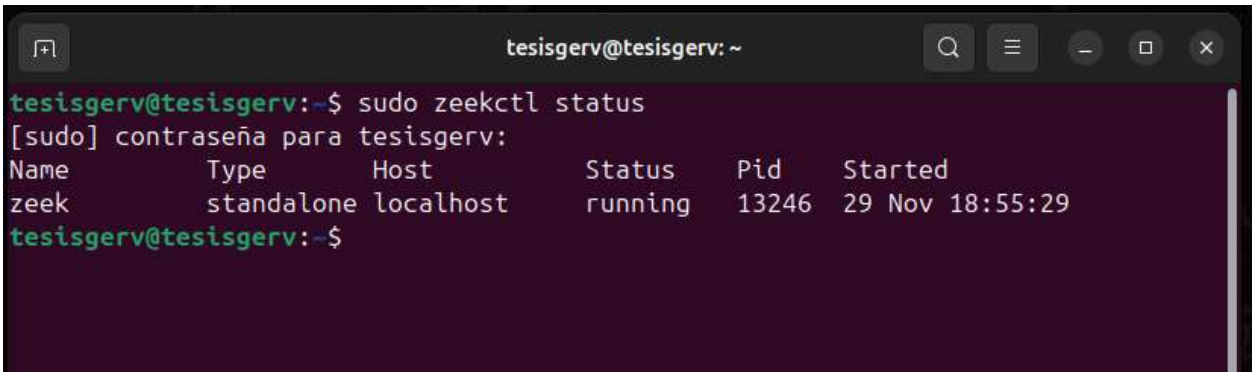


Figura 35 - Zeek Status = Running

Este comando permite verificar el estado operativo del sensor Zeek como se muestra en la *Figura 45*. Los resultados indican que:

- ✓ El servicio está ejecutándose correctamente (running).
- ✓ Zeek trabaja en modo standalone, es decir, funciona como un único nodo encargado de capturar y analizar el tráfico.

- ✓ El proceso se encuentra activo bajo el PID correspondiente y con fecha/hora del último arranque.

Esto confirma que el sensor se encuentra en funcionamiento estable y listo para la captura continua de tráfico.

2. Activación del monitoreo en la interfaz de red

Zeek detecta tráfico directamente desde la interfaz configurada:

cat /opt/zeek/etc/node.cfg

```
tesisgerv@tesisgerv:~$ cat /opt/zeek/etc/node.cfg
[zeek]
type=standalone
host=localhost
interface=ens33
```

Figura 36 - Archivo node.cfg de Zeek

Al ejecutar el comando se muestra los archivos que define los parámetros principales del nodo de monitoreo, La *Tabla 9* muestran las configuraciones relevantes.

Tabla 9 - Configuraciones Relevantes del sensor Zeek

PARÁMETROS PRINCIPALES	ESTADO	FUNCIÓN
type	standalone	Indica que se utilizará un único sensor Zeek.
host	localhost	El sensor está instalado y operando en la misma máquina.
interface	ens33	Determina que Zeek captura tráfico únicamente en la interfaz ens33

Esto garantiza que Zeek está recibiendo el tráfico correcto desde la interfaz monitoreada y que su configuración está alineada con los demás sensores del proyecto.

3. **Verificación de generación de logs (directorio current/)**

Al ejecutar:

ls -lh /opt/zeek/logs/current/

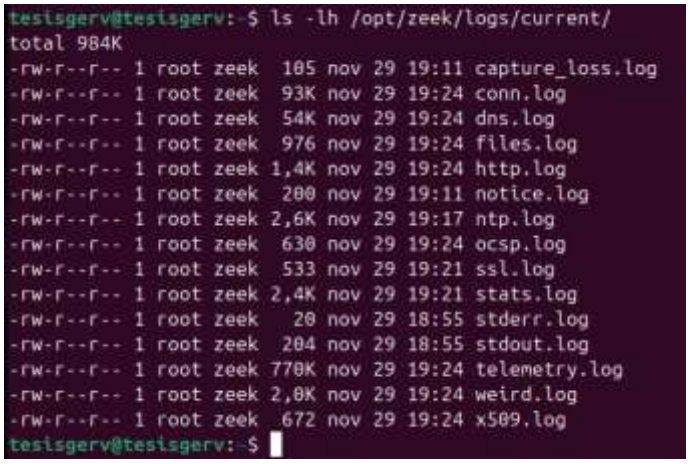


Figura 37 - Directorio current de Zeek

Este directorio contiene los registros generados por el sensor durante su funcionamiento. La *Figura 47* evidencia que Zeek está generando de forma correcta los archivos principales las cuales su función es mostrada en la *Tabla 10.*

Tabla 10 – Archivos Principales detectados por Zeek

ARCHIVOS LOGS	PROTOCOLO	FUNCIONAMIENTO
conn.log	Capa TCP/UDP	Registro de todas las conexiones de red detectadas.
dns.log	DNS	Consultas DNS realizadas.
http.log	HTTP	Actividad HTTP, incluyendo métodos, URL y encabezados.
ssl.log	TLS/SSL	Información sobre conexiones TLS/SSL
files.log	FTP	Ficheros transferidos o detectados en el tráfico
weird.log	HTTP/SSL/DNS	Eventos anómalos no identificados por protocolos estándar.
notice.log	SSL/TLS/SSH	Alertas y eventos relevantes generados por scripts o políticas.

La presencia y actualización de estos archivos confirma:

- ✓ Que Zeek está procesando tráfico real.
- ✓ Que el sensor está capturando múltiples protocolos.
- ✓ Que el entorno de monitoreo se encuentra funcionando de manera adecuada.

Estos registros serán utilizados posteriormente para el análisis de tráfico, detección de patrones y correlación con los eventos generados por Suricata.

4. Visualización del contenido de los logs

Después de confirmar que Zeek está instalado, activo y capturando tráfico correctamente como se evidencia en las capturas previas con los comandos *zeekctl status* y el contenido de */opt/zeek/logs/current/,* procedemos a visualizar lo que el sensor IDS está detectando en tiempo real.

Zeek genera múltiples archivos de log, cada uno especializado en un protocolo o tipo de información. Para fines del proyecto, los logs analizados fueron:

- ✓ conn.log = Conexiones de red detectadas
- ✓ http.log = Tráfico HTTP en texto claro
- ✓ ssl.log = Sesiones TLS/SSL

5. conn.log - Registro de todas las conexiones detectadas

sudo tail -n 20 /opt/zeek/logs/current/conn.log

Este archivo contiene un registro detallado de cada conexión TCP, UDP e ICMP detectada por Zeek como se muestra en la *Figura 48*.

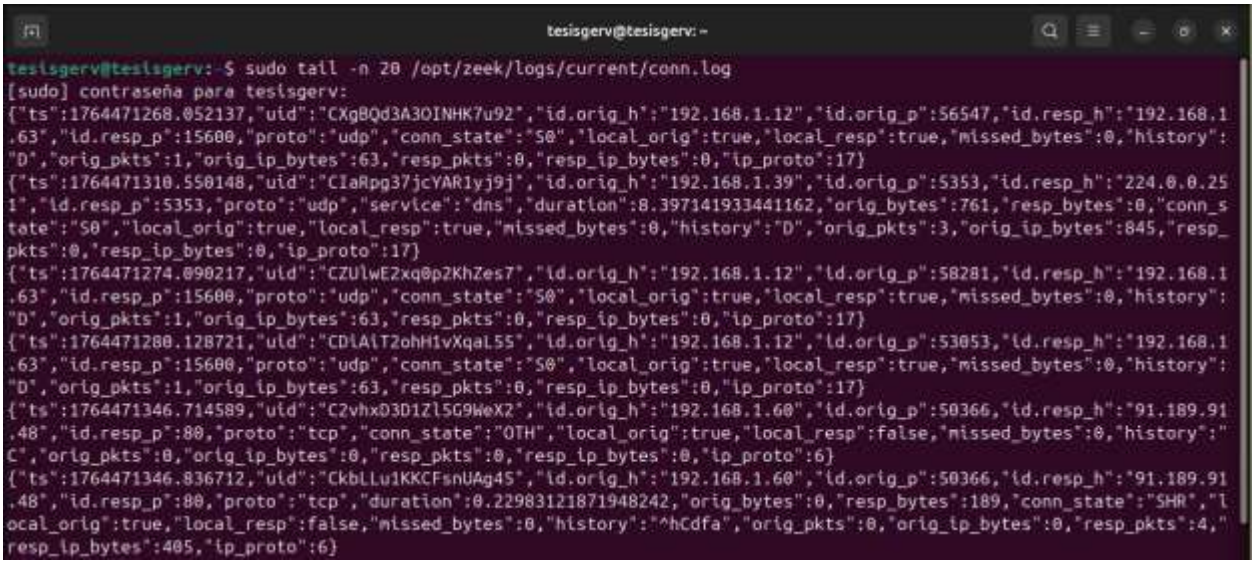


Figura 38 – Registro ultimas conexiones procesadas conn.log

Tabla 11 – Interpretación de campos – conn.log

CAMPO	SIGNIFICADO	UTILIDAD EN IDS
`ts`	Timestamp del evento	Saber cuándo ocurrió la conexión
`uid`	Identificador único de la conexión	Permite correlación entre logs
`id.orig_h`	IP de origen	Identifica quién generó la conexión
`id.orig_p`	Puerto de origen	Útil para detectar patrones de escaneo
`id.resp_h`	IP destino	Saber hacia dónde se dirigía el tráfico
`id.resp_p`	Puerto destino	Detectar servicios atacados (22, 80, 443, etc.)
`proto`	Protocolo (tcp/udp/icmp)	Clasifica el tráfico
`service`	Protocolo de aplicación identificado	HTTP, DNS, SSL, etc.
`duration`	Duración de la conexión	Conexiones largas pueden ser exfiltración
`orig_bytes / resp_bytes`	Bytes enviados/recibidos	Detecta picos o descargas grandes
`conn_state`	Estado de la conexión	Muy útil para detectar anomalías
`local_orig/resp`	Indica si IP está en red interna	Distingue tráfico interno vs externo

Es uno de los logs más importantes porque resume la actividad general de red como se muestra en la Tabla 11 y permite identificar:

- ✓ Conexiones sospechosas
- ✓ Escaneos de puertos
- ✓ Tráfico hacia direcciones desconocidas
- ✓ Fallos de handshake
- ✓ Paquetes mal formados o repetidos

6. http.log - Registro de tráfico HTTP

Para ver eventos HTTP ejecutamos:

sudo tail -n 20 /opt/zeek/logs/current/http.log

Figura 39 – Registro de tráfico HTTP – http.log

Este log registra interacciones HTTP en texto claro sin ningún cifrado para entenderlos revisamos la *Tabla 12*. Los valores que se muestran en la *Figura 49* la pantalla indica la comunicación entre la PC_Server IDS y servidores externos.

Tabla 12 – Interpretación de campos – http.log

CAMPO	SIGNIFICADO	UTILIDAD
`ts`	Momento de la solicitud HTTP	Permite correlación con Suricata
`uid`	ID correlacionado con 82on.log	Une eventos de distintos logs
`id.orig_h`	IP del cliente	Quién hizo la petición
`id.resp_h`	IP del servidor	A dónde se conectó
`method`	GET / POST, etc.	Detectar acceso sospechoso
`host`	Dominio solicitado	Ver webs consultadas
`uri`	Ruta dentro del sitio	Detectar patrones maliciosos
`status_code`	Código HTTP	Errores 400/500 pueden ser ataques
`user_agent`	Cadena del navegador	SOLO si no está cifrado
`tags`	Indicadores de Zeek	Análisis adicional

7. ssl.log – Registro de sesiones TLS

Al ejecutar:

sudo tail -n 20 /opt/zeek/logs/current/ssl.log

Figura 40 – Últimos Registros TLS – ssl.log

Este log muestra cada sesión HTTPS establecida.

Es clave para monitorear:

- ✓ Qué servidores reciben tráfico cifrado
- ✓ Versiones TLS
- ✓ Cipher Suites
- ✓ Posibles intentos de downgrade
- ✓ Certificados sospechosos

Tabla 13 – Interpretación de campos – ssl.log

CAMPO	SIGNIFICADO	UTILIDAD
ts	Timestamp	Correlación temporal
uid	ID correlacionado con conn.log	Unión de eventos
version	Versión TLS	Detectar TLS obsoleto
cipher	Algoritmo de cifrado	Ver si usan cifrados inseguros
curve	Curva ECC (si aplica)	Seguridad criptográfica
resumed	Sesión reusada	Indicador de comportamiento normal
established	Si completó handshake	Fallos indican ataques
ssl_history	Eventos del handshake	Análisis detallado

8. Creación de script propio: detección de User-Agent (lab-http-ua.zeek)

Este script fue desarrollado con el objetivo de extender las capacidades nativas de Zeek, permitiendo la detección de un User-Agent específico utilizado en las pruebas de este proyecto. Debido a que el tráfico HTTP actual viaja mayoritariamente cifrado (HTTPS), Zeek únicamente

registra User-Agent cuando el tráfico es HTTP plano. Por ello, se incorporó un script adicional capaz de inspeccionar el header HTTP antes de que sea almacenado en http.log.

Este script detecta cualquier petición HTTP cuyo User-Agent contenga el texto determinado.

Al ejecutar:

/opt/zeek/share/zeek/site/lab-http-ua.zeek

Encontraremos la ubicación del script. Al entrar a ese código nos refleja una pantalla donde crearemos el script personalizado como se muestra en la *Figura 46*.

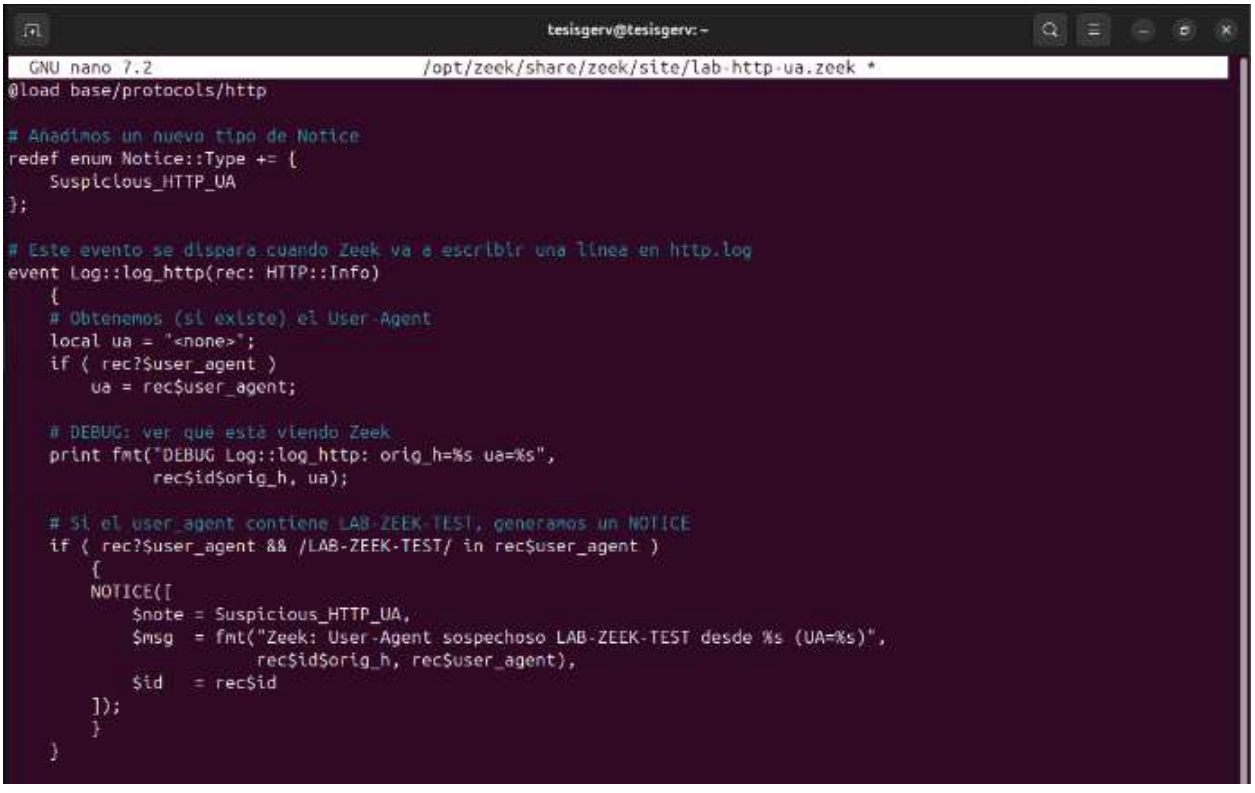


Figura 41 - Patrón definido por el laboratorio "LAB-ZEEK-TEST"

El script intercepta cada evento HTTP extrae el campo User_Agent cuando está disponible y compara su contenido con un patrón definido por el laboratorio "LAB-ZEEK-TEST".

3.3.3 Integración Suricata + Zeek en entorno mixto

En este proyecto se implementó un IDS mixto donde Suricata y Zeek trabajan en paralelo analizando el mismo tráfico proveniente del laboratorio. Cada motor aporta capacidades distintas:

- Suricata: análisis basado en reglas (signature-based), detección rápida de patrones, alertas directas.
- Zeek: análisis comportamental y registro profundo por protocolo (behavior-based).

Esta combinación mejora la visibilidad y permite correlacionar eventos.

1. Flujo de tráfico desde el laboratorio físico hacia el IDS

El tráfico generado dentro del laboratorio como lo es PC_Server IDS, dispositivos IoT y el WiFi Pineapple utilizado en las pruebas controladas, es enviado hacia los sensores Suricata y Zeek mediante un proceso de duplicación de paquetes.

En la infraestructura física *Figura 14*, el switch MikroTik CRS310 estaba destinado a realizar Port Mirroring que es la monitorización de puertos para replicar todo el tráfico proveniente de los segmentos del laboratorio hacia la interfaz de captura del servidor IDS.

Al ejecutar los siguientes comandos validamos el flujo en el servidor IDS:

- Para ver interfaz activa *ip addr show ens33* como se muestra en la *Figura 47*

```
tesisgerv@tesisgerv:~$ ip addr show ens33
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:60:10:4f brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.1.30/26 brd 192.168.1.63 scope global dynamic noprefixroute ens33
       valid_lft 77412sec preferred_lft 77412sec
```

Figura 42 - Interfaz Activa

- Verificar si Suricata y Zeek están capturando tráfico en la misma interfaz:

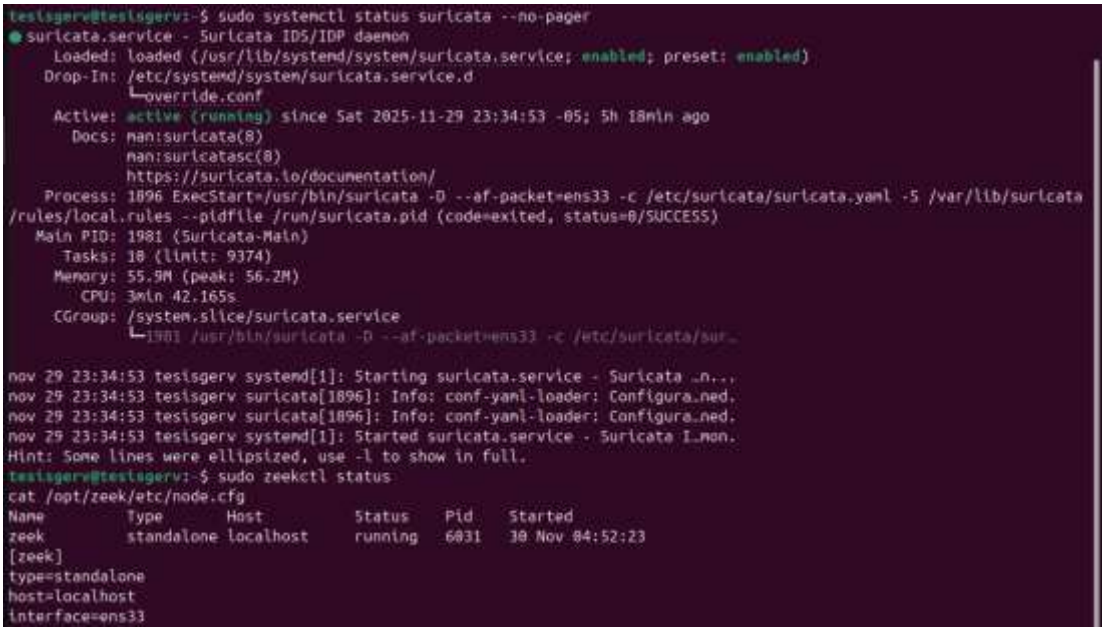


Figura 43 – Los dos sensores están capturando Tráfico en la misma interfaz

- Verificación de paquetes en tiempo real ejecutamos el siguiente comando *sudo tcpdump -i ens33 -nn -c 20:*

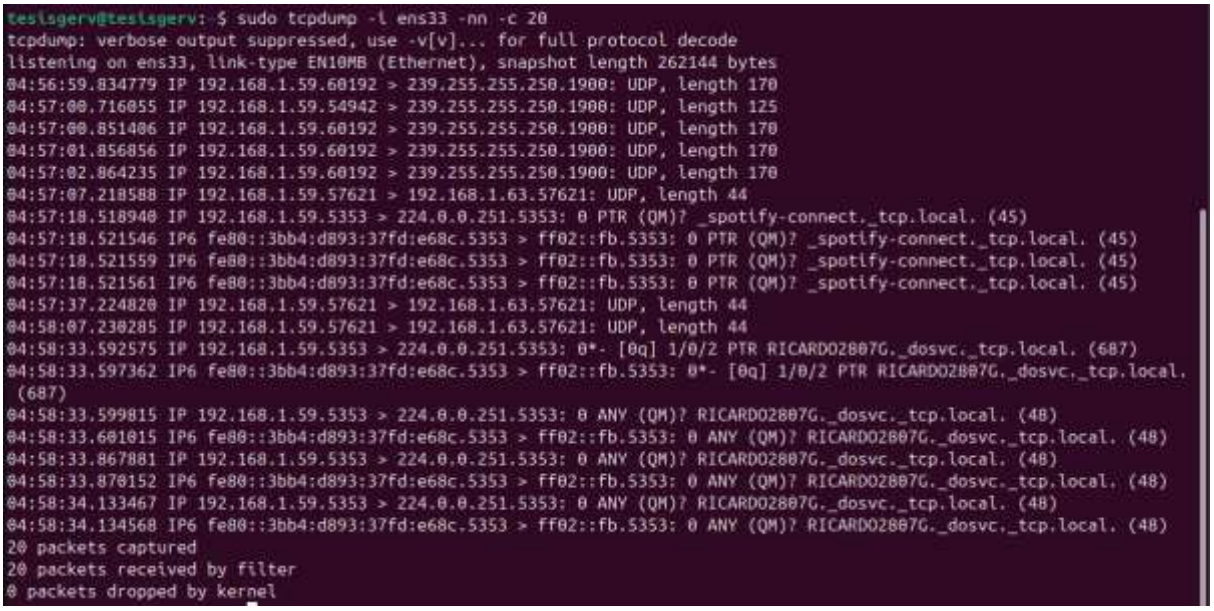


Figura 44 - Demostración de paquetes en tiempo real

Debido a limitaciones de acceso al laboratorio, esta funcionalidad se replicó en el entorno virtual utilizando las interfaces de VMware, permitiendo que ambos motores como lo son Suricata y Zeek reciban exactamente la misma carga de tráfico para análisis simultáneo.

2. Sincronización de logs para análisis posterior

Para garantizar la correlación entre eventos detectados por Suricata y Zeek, se estandarizó la captura a nivel de la misma interfaz ens33 y se verificó que ambos motores registraran el tráfico en tiempo real.

- Suricata genera logs estructurados (JSON) que son consumidos fácilmente por Loki y Grafana.
- Zeek en cambio produce archivos independientes por cada protocolo (conn.log, http.log, dns.log, ssl.log).

Al ejecutar los siguientes códigos observamos en la *Figura 50*, el comportamiento de los dos sensores en la misma conexión en su demostración real:

Figura 45 -Demostración real de los dos sensores y sus funciones

Tabla 14 – Archivos de Logs utilizados

Sensor	Archivo	Descripción
Suricata	eve.json	Log estructurado principal: alertas, http, dns, tls, flow
Suricata	fast.log	Alertas rápidas en texto plano
Zeek	conn.log	Todas las conexiones TCP/UDP
Zeek	http.log	Métodos, User-Agent, hostnames (solo HTTP plano)
Zeek	dns.log	Consultas y respuestas DNS
Zeek	ssl.log	Información de sesiones TLS

La sincronización se consigue porque ambos procesan el mismo flujo de paquetes al mismo tiempo, permitiendo comparar. Un User-Agent sospechoso detectado por Suricata con la información de la misma conexión registrada en Zeek mediante su conn.log y http.log.

3.4 Implementación del Sistema de Recolección y Almacenamiento de Logs

En el sistema IDS híbrido del laboratorio en unidad a Suricata como Zeek generan un volumen significativo de logs con información crítica del tráfico. Para centralizar estos registros y permitir su consulta desde un único punto, se implementó un stack de observabilidad basado en Grafana Loki + Promtail, actuando como un Data Lake de seguridad.

3.4.1 Arquitectura Loki-Promtail

La arquitectura adoptada se basa en un modelo “push-based”, donde los agentes Promtail instalados en el servidor IDS recolectan logs locales y los envían a Loki mediante HTTP.

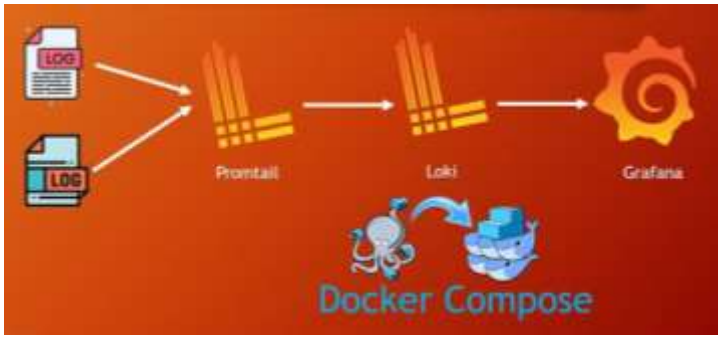


Figura 46 – Comportamiento de una Arquitectura Loki-Promtail [36]

Esta arquitectura resuelve tres necesidades del proyecto:

- Recolección unificada de logs generados por Suricata y Zeek.
- Envió estructurado de logs con etiquetas declarada labels esta para facilitar las búsquedas.
- Almacenamiento optimizado para eventos de seguridad donde Loki actúa como Data Lake.

Tabla 15 - Componentes para el Funcionamiento del Sistema IDS

COMPONENTE	FUNCIÓN DENTRO DEL SISTEMA IDS
Suricata	Genera logs estructurados: eve. json, fast.log, stats.log
Zeek	Genera logs por protocolo: conn.log, dns.log, http.log, ssl.log, entre otros
Promtail	Agente local que lee ambos grupos de logs y les añade etiquetas (origen, servicio, severidad, etc.)
Loki	Almacena los logs como flujos indexados
Grafana	Panel de visualización y correlación de alertas

Promtail funciona como un colector inteligente, que detecta nuevos eventos en los ficheros, extrae metadatos y los envía a Loki en tiempo real.

3.4.2 Configuración de Promtail

Para la recolección de logs se utilizó */etc/promtail/config.yml.*

El objetivo de esta configuración fue incluir de forma simultánea los logs más relevantes de Suricata y Zeek.

Para verificar Promtail en el entorno del IDS, revisamos los contenedores activos ejecutamos el siguiente código:

docker ps --format "table {{.Names}}\t{{.Status}}\t{{.Ports}}"

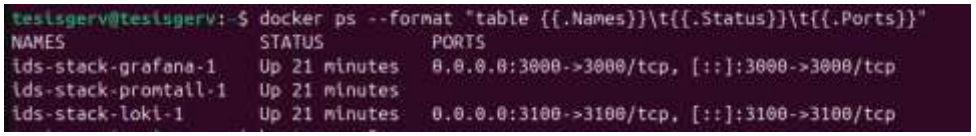


Figura 47 - Contenedores Activos

Revisamos los logs del contenedor Promtail ejecutando *docker logs ids-stack-promtail-1 --tail 50:*

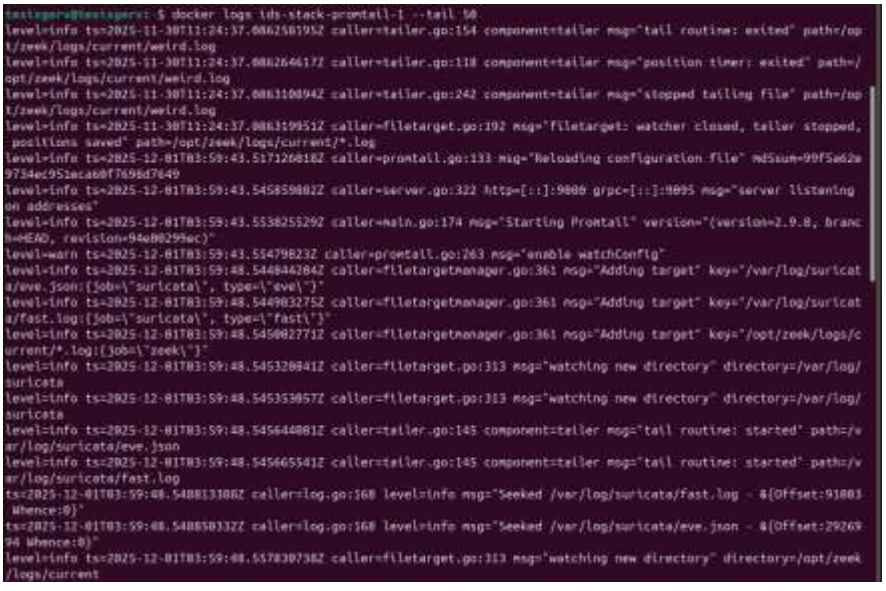


Figura 48 - Logs generados en el contenedor de Promtail

- **Verificamos la configuración de Promtail**

Ejecutamos el siguiente comando *docker exec -it ids-stack-promtail-1 sh*, una vez dentro se ejecuta otro código la cual es *cat /etc/promtail/config.yml* como se muestra en la *Figura 54*.

```
tesisgerv@tesisgerv:~$ docker exec -it ids-stack-promtail-1 sh
# cat /etc/promtail/config.yml
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
  - url: http://loki:3100/loki/api/v1/push

scrape_configs:
- job_name: system
  static_configs:
  - targets:
      - localhost
    labels:
      job: varlogs
      __path__: /var/log/*log
#
```

Figura 49 – Configuraciones de Promtail

En esta Figura nos muestra:

- ✓ Las rutas de Suricata
- ✓ Las rutas de Zeek
- ✓ Los labels configurados
- ✓ La configuración del cliente Loki (URL: loki:3100)

- **Rutas de logs de Suricata**

Suricata genera múltiples logs en /var/log/suricata/.

Tabla 16 – Rutas incorporadas en Promtail

Log	Ruta	Función
eve.json	`/var/log/suricata/eve.json`	Log estructurado JSON con alertas, HTTP, DNS, TLS, flows
fast.log	`/var/log/suricata/fast.log`	Alertas rápidas en formato legible
stats.log	`/var/log/suricata/stats.log`	Estadísticas de rendimiento del IDS

Estos logs permiten reconstruir el tráfico y detectar alertas de Suricata.

- **Rutas de logs de Zeek**

Zeek genera logs divididos por protocolos dentro de:

/opt/zeek/logs/current/

Tabla 17 – Principales logs recolectados

Log	Ruta	Descripción
conn.log	/opt/zeek/logs/current/conn.log	Detalles de cada conexión: IPs, puertos, protocolo, duración
http.log	/opt/zeek/logs/current/http.log	Información del tráfico HTTP (método, URL, headers)
dns.log	/opt/zeek/logs/current/dns.log	Consultas y respuestas DNS
ssl.log	/opt/zeek/logs/current/ssl.log	Handshakes TLS, versiones, ciphers negociados
weird.log	/opt/zeek/logs/current/weird.log	Eventos anómalos detectados por Zeek

3.4.3 Configuración de Loki como Data Lake de seguridad

Loki actúa como un Data Lake centralizado donde confluyen todos los registros generados por Suricata y Zeek.

A diferencia de sistemas tradicionales que almacenan logs como texto plano, Loki utiliza un modelo basado en etiquetas (labels), lo que permite consultas rápidas y eficientes desde Grafana sin necesidad de indexar todo el contenido de cada línea.

En esta implementación, Loki recibe logs a través de Promtail utilizando el endpoint:

http://loki:3100/loki/api/v1/push

Se validó que Loki esté activo dentro del stack Docker y que recibe correctamente los streams enviados por Promtail. La arquitectura permite mantener los logs centralizados, accesibles y consultables en tiempo real para análisis de seguridad y correlación de eventos IDS.

Loki no indexa el contenido completo, sino solo labels, lo cual esto reduce consumo de RAM y CPU. Esto aumenta la velocidad de consulta y mantiene una estructura “append-only” ideal para logs de IDS. Esto permite realizar búsquedas como:

{job="suricata", event_type="alert"}

3.5 Implementación de Sistema de Visualización y Monitoreo

La última etapa del despliegue de la arquitectura IDS híbrida consiste en integrar todos los registros recolectados por Suricata y Zeek dentro de una plataforma visual que permita supervisar, analizar y correlacionar eventos de seguridad en tiempo real.

Para ello se implementó Grafana, una herramienta ampliamente utilizada en entornos SOC (Security Operations Center) para el monitoreo centralizado de datos provenientes de diversas fuentes.

3.5.1 Configuración de Grafana

En esta fase se habilitó Grafana como interfaz principal para consultar los logs recolectados por Loki. Se configuró Loki como Data Source como se muestra en la *Figura 52*, lo que permite consultar, filtrar y visualizar los datos enviados por Promtail desde Suricata y Zeek.

- ✓ Acceso a la Interfaz Web de Grafana

En el navegador de mi VM buscamos de la siguiente manera, agregando el *http://192.168.1.30:3000* como se muestra en la *Figura 55*:

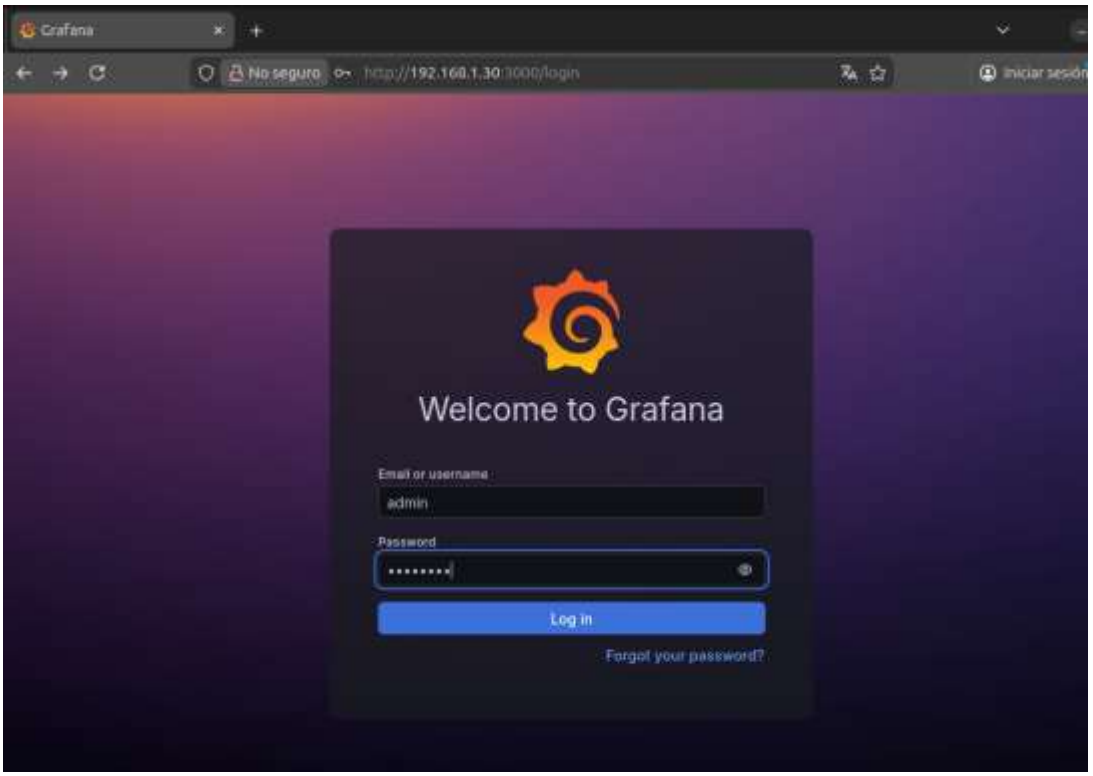


Figura 50 - Pantalla de login Grafana

✓ Configuración de Loki como Data Source

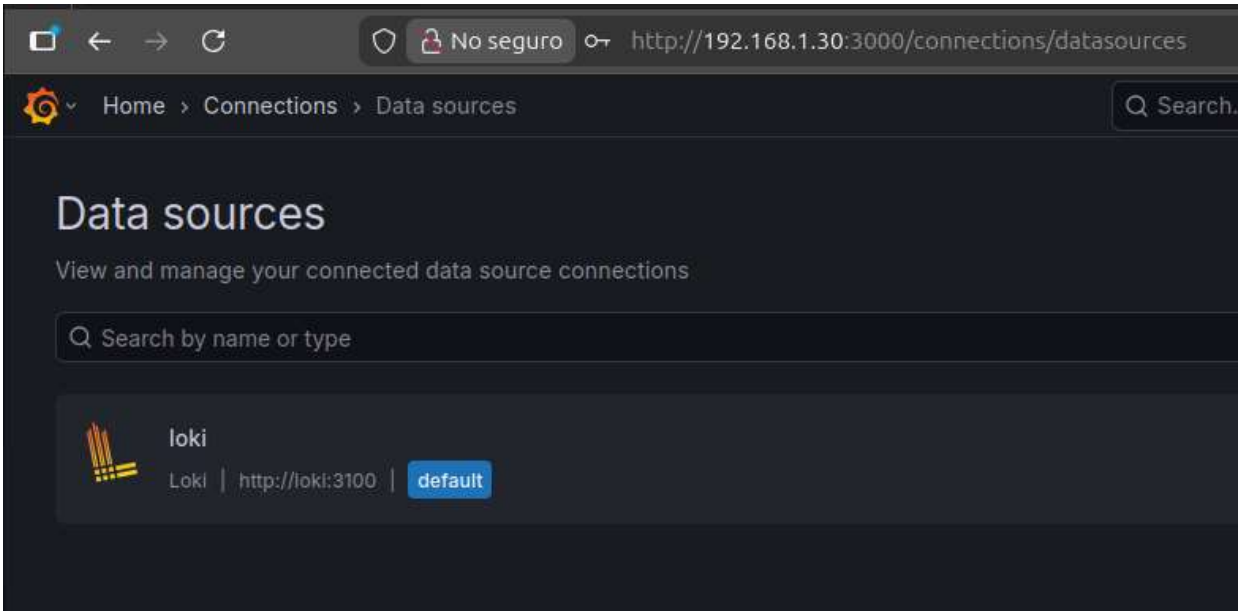


Figura 51 - Loki - Data source is working

Tabla 18 - Parámetros configurados en la Data Source Loki

PARÁMETRO	VALOR ASIGNADO	FUNCIÓN
URL	`http://localhost:3100`	Dirección interna del servicio Loki
Max series	1000	Límite de streams consultados
Timeout	30s	Tiempo máximo de espera de consulta
Query editor	Loki Query Builder	Interfaz para consultas LOKI

3.5.2 Diseño de Paneles de Visualización

El diseño de los paneles en Grafana se orientó a que el operador pueda identificar de forma rápida qué está ocurriendo en la red del laboratorio y en qué componente del IDS se genera cada evento. Para ello se organizaron los paneles en tres grupos: uno centrado en Suricata, otro en Zeek y un panel resumen del sistema IDS.

Los paneles se diseñaron siguiendo estas secciones:

Fila 1 – Actividad Global del IDS Panorama General

- ✓ **Panel 1 – Comparación de Actividad IDS: Suricata vs Zeek (Eventos por Minuto)**

Este panel compara la actividad registrada por Suricata y Zeek dentro del mismo intervalo temporal, permitiendo contrastar la detección basada en firmas con el análisis de comportamiento. Su funcionalidad es mostrar cómo ambos sensores aportan perspectivas complementarias, enriqueciendo la observación del tráfico de red. Se utiliza porque la correlación entre sensores incrementa la precisión del monitoreo, permitiendo validar eventos, detectar inconsistencias y obtener una visión integral del comportamiento de la red dentro del laboratorio IDS.

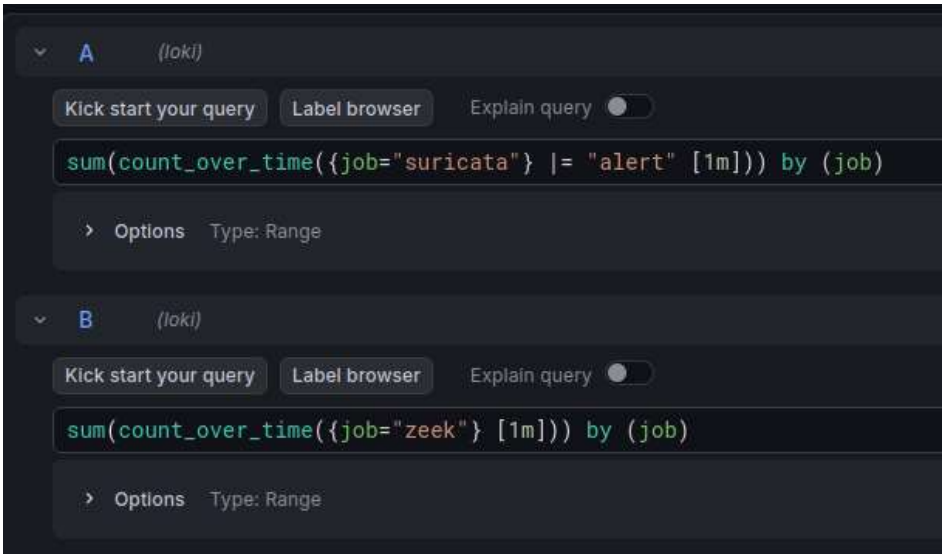


Figura 52 - Ejecución del código consultando en Loki para la creación del Panel 1

- ✓ **Panel 2 – Tendencia de Alertas IDS (Tiempo Real)**

Este panel muestra la evolución temporal de las alertas generadas, revelando picos, disminuciones y patrones de comportamiento a lo largo del tiempo. Su funcionalidad es permitir

el análisis dinámico del tráfico, mostrando cuándo se producen concentraciones de actividad sospechosa. Se utiliza porque ayuda a relacionar eventos con momentos específicos, facilitando la correlación con actividades internas, pruebas de laboratorio o posibles ataques. Su enfoque temporal aporta claridad sobre el ritmo y magnitud de los incidentes detectados.

Figura 53 - Ejecución del código consultando en Loki para la creación del Panel 2

Fila 2 – Clasificación de Alertas

- ✓ **Panel 3 – Distribución por Alertas por Protocolo (Suricata)**

Este panel agrupa las alertas según el protocolo de red involucrado, proporcionando una representación clara de qué tipo de tráfico origina mayor cantidad de eventos. Su funcionalidad es identificar patrones relacionados con protocolos específicos intentos de escaneo mediante ICMP o conexiones anómalas en puertos TCP y UDP. Se utiliza porque permite comprender mejor la superficie de ataque del entorno, facilitando la detección temprana de actividades inusuales que suelen concentrarse en ciertos protocolos mostrados en la *Figura 64*.

Figura 54 - Ejecución del código consultando en Loki para la creación del Panel 3

✓ Panel 4 - Clasificación de Alertas por Severidad (Suricata)

Este panel presenta la distribución de alertas clasificadas por severidad generadas por el sensor Suricata en tiempo real. En la *Figura 65* Suricata asigna a cada alerta un nivel de severidad basado en la criticidad del evento

El panel permite identificar rápidamente si el entorno presenta actividad potencialmente peligrosa, al visualizar la proporción entre eventos leves y críticos. De esta manera se facilita la priorización de incidentes durante el monitoreo del laboratorio IDS.

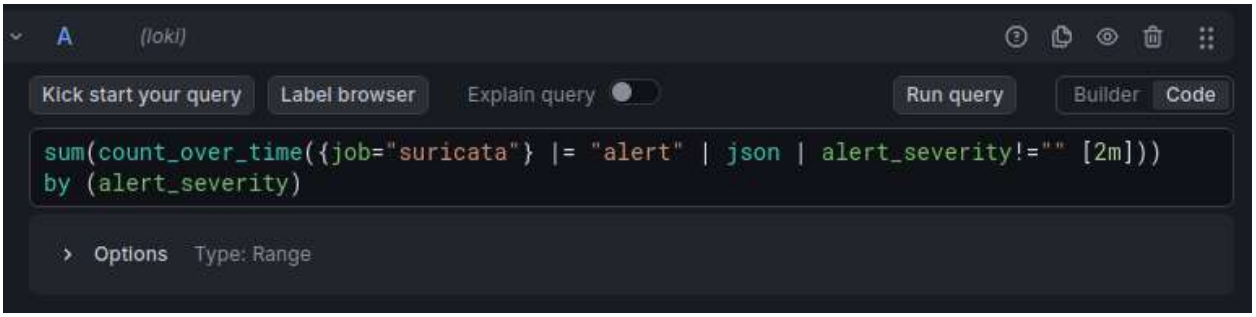


Figura 55 - Ejecución del código consultando en Loki para la creación del Panel 4

✓ Panel 5 - Conteo Total de Alertas IDS (Suricata)

Este panel muestra las direcciones IP que generan la mayor cantidad de alertas, permitiendo identificar rápidamente los principales orígenes de actividad sospechosa *Figura 65*.

Su funcionalidad es destacar hosts potencialmente comprometidos o dispositivos que intentan interactuar de manera indebida con la red monitoreada. Se utiliza porque conocer la fuente de los eventos facilita el aislamiento, la mitigación y la toma de decisiones informadas, convirtiéndose en un elemento clave para la respuesta ante incidentes dentro del entorno del laboratorio.

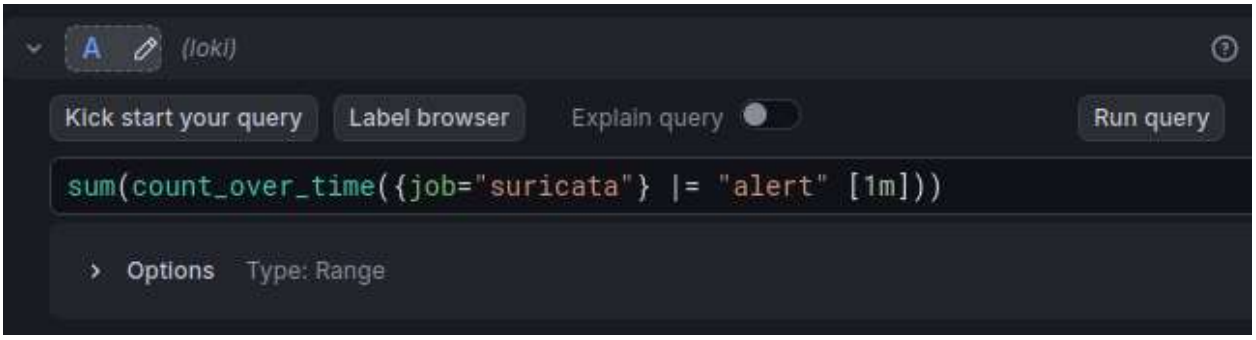


Figura 56 - Ejecución del código consultando en Loki para la creación del Panel 4

✓ Panel 5 – Top Firmas IDS Más Frecuentes

Este panel presenta las firmas de alerta que Suricata detecta con mayor recurrencia, permitiendo identificar qué tipos de amenazas o comportamientos anómalos se repiten en el entorno evaluado. Su funcionalidad es resaltar patrones de ataque específicos, como escaneos, intentos de explotación o solicitudes maliciosas.

Se utiliza porque conocer las firmas más frecuentes ayuda a comprender el contexto del tráfico y a evaluar si existen campañas persistentes o comportamientos repetitivos que requieran una intervención más profunda como en la *Figura 66*.

Figura 57 - Ejecución del código consultando en Loki para la creación del Panel 5

3.6 Implementación del Sistema de Notificación IDS

La presente sección describe el proceso de integración, configuración y desarrollo del sistema de notificaciones implementado para el SERVER_IDS, el cual opera como un componente central dentro del Laboratorio de Telecomunicaciones de la UPSE. El objetivo principal es contar con un mecanismo automatizado capaz de alertar en tiempo real sobre eventos sospechosos detectados por los sensores Suricata y Zeek, enviando notificaciones directamente a un canal privado del administrador mediante la API de Telegram.

3.6.1 Configuración del Sistema de Notificaciones con Telegram

El servicio de mensajería Telegram fue seleccionado por su rapidez, bajo consumo de recursos, soporte para API abiertas y facilidad para integrarse con sistemas de monitoreo. Su uso permite que el administrador del laboratorio reciba alertas en tiempo real para posterior a eso el monitoreo con el Dashboard Grafana.

- **Creación del Bot de Telegram**

Para habilitar el canal de comunicación se configuró un bot a través de BotFather como se muestra en la *Figura 69*, obteniéndose:

  - **BOT_TOKEN:** Credencial única para enviar mensajes mediante la API HTTPS.
  - **CHAT_ID:** Identificador del canal privado donde se reciben las alertas.

Este mecanismo brinda seguridad, ya que únicamente el administrador autorizado puede visualizar los eventos IDS generados.

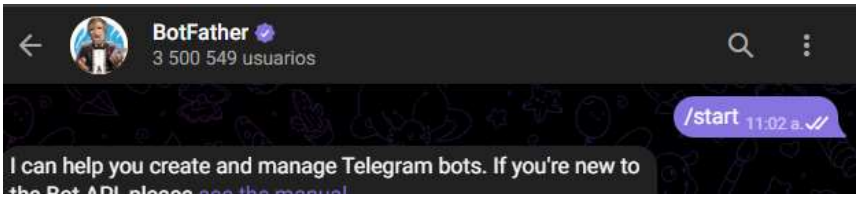


Figura 58 - BotFather Telegram

- **Integración con Equipos MikroTik**

En la planificación inicial del proyecto se contempló utilizar el router MikroTik CRS310/RB4011 como generador de notificaciones mediante su módulo scripting, aprovechando herramientas como:

Tabla 19 – Herramientas para modulaciones de Script

HERRAMIENTA	FUNCIÓN
/tool fetch	Para peticiones HTTPS
/system	Script para automatizar mensajes
/log print follow	Para capturar anomalías.

Se decidió migrar completamente la capa de notificaciones al SERVER_IDS MV Ubuntu, manteniendo al MikroTik exclusivamente como dispositivo de tráfico espejo conocido como el port mirroring hacia los sensores IDS.

Esta decisión garantizó:

- Funcionamiento estable.
- Comunicación 100% funcional con Telegram.
- Capacidad de ejecutar scripts más complejos no soportados por RouterOS.

- **Validación de comunicación desde Ubuntu**

En el servidor IDS se verificó la salida hacia Telegram mediante:

- *curl -v https://api.telegram.org/botTOKEN/getMe*

Figura 59 – Verificación de salida de Telegram para el enlace con el Bot

La conexión entre el servidor IDS y la API de Telegram se validó mediante una consulta GET con cURL como se muestra en la *Figura 70*, garantizando que el bot estuviera correctamente registrado y operativo antes de integrar el sistema automatizado de notificaciones.

Esto para obtener una respuesta exitosa confirmando que el servidor podía transmitir alertas sin restricciones y sin depender del MikroTik bloqueado por políticas de red.

3.6.2 Desarrollo del Script de Alertas Automatizadas

El corazón del sistema es un script unificado en Python3 que integra los dos sensores IDS utilizados en el proyecto Suricata y Zeek. Su desarrollo se realizó siguiendo criterios de modularidad, eficiencia, resiliencia y claridad en el procesamiento de eventos.

Para crear el script **ids_alert_bot.py** se creó una carpeta que se encuentra alojado en:

- */usr/local/bin/ids_alert_bot.py*

siendo ejecutado automáticamente como servicio mediante systemd, será donde se almacene todo el comportamiento del script.

- ✓ **Estructura General del Script**

El script se encuentra organizado por bloques funcionales, cada uno con un propósito claro dentro del sistema IDS. La siguiente descripción explica qué hace cada sección, por qué existe y cuál es su rol en la automatización del monitoreo.

1. **Línea Shebang – Identificación del Intérprete**

   - *#! /usr/bin/env python3*

Es una línea especial que le indica al sistema operativo con qué intérprete debe ejecutarse al script garantizando que se ejecute sin ningún problema con Python 3 siendo este independientemente de la ubicación exacta del binario en el servidor.

Figura 60 – Línea Shebang y las importaciones como cabeza del Script

Estos apartados mostrados en la *Figura 71* asegura la compatibilidad y evita errores al levantar el servicio automático systemd. Sin esta línea el script podría no ejecutarse correctamente como servicio.

2. Importación de Librerías

Es el bloque donde se cargan módulos de Python que el script necesita. Es fundamental dentro del proyecto porque nos permite la comunicación entre los sensores Suricata y Zeek, junto al Servidor IDS y el Telegram.

Estas librerías que se interpreta en la *Tabla 20,* son el sistema para las alertas y poder enviarlas:

Tabla 20 – Función de librería en el Script

LIBRERÍA	FUNCIÓN
json	Interpretar los archivos *eve. json* de Suricata y *notice.log* de Zeek
time	Manejar pausas, ciclos y evitar saturación del CPU
requests	Enviar las alertas al API de Telegram
os	Verificar tamaño y existencia de archivos
datetime	Controlar ventanas de tiempo para evitar alertas repetidas en ICMP

3. Configuración del Bot

Es la configuración clave que permite comunicar el servidor IDS con Telegram.

```
# ========================
# CONFIGURACIÓN DEL BOT
# ========================
TOKEN = "8596314391:AAH6Vlu2fiFiCIgEelrswgWVkQGddfF--kk"
CHAT_ID = "1966250935"
TELEGRAM_URL = f"https://api.telegram.org/bot{TOKEN}/sendMessage"
```

Figura 61 – Datos únicos de Bot para vincular Server y Telegram son definidas como variables

La utilización de estas condiciones mostradas en la *Figura 72* nos indica que Telegram funciona mediante una API la cual necesita de:

- ✓ TOKEN del bot = Identifica quién envía los mensajes
- ✓ CHAT_ID = A qué usuario o grupo se enviarán
- ✓ URL del API = Dirección donde el script enviará las alertas

Tiene como función en el proyecto en transforma alertas de IDS en notificaciones automáticas, permitiendo monitoreo remoto. Sin esta sección, no existiría comunicación con Telegram.

4. Rutas de archivos Suricata y Zeek

En este apartado nos indica que es la declaración de las rutas donde Suricata y Zeek generan sus alertas.

```
# ============================================================
# RUTAS DE ARCHIVOS DE IDS
# ============================================================
SURICATA_EVE = "/var/log/suricata/eve.json"
ZEEK_NOTICE = "/opt/zeek/spool/zeek/notice.log"
```

Figura 62 – Direcciones de registro donde se encuentran todas configuraciones de cada sensor

El script necesita leer los eventos brutos generados por los IDS. La función dentro del proyecto es trabajar con las reglas que se personalizaron, esto dándole al bot una fuente de información para detectar ataques o anomalías.

Define las dos fuentes oficiales de seguridad del laboratorio:

- ✓ Suricata = Eventos de red y ataques basados en firmas

- ✓ Zeek = Análisis comportamental y anomalías

5. Mapeo de Severidades

Este apartado consiste en tener una tabla interna que traduce valores numéricos a niveles de severidad con significado visual. Los IDS reportan la severidad en números, pero eso no es comprensible para un operador humano. Así que se la declaro como se muestra en la Figura

Figura 63 - Definición por Rango y colores según su tipo de trafico

Su función es de permite que los mensajes enviados a Telegram sean claros, visuales, interpretables de inmediato. Esto mejora la velocidad de reacción del analista frente a un incidente.

6. Función para enviar mensajes a Telegram

Esta función encapsula el envío de mensajes, evitando repetir código y centraliza el envío de alertas, garantizando que cada evento importante sea notificado al encargado de monitoreo.

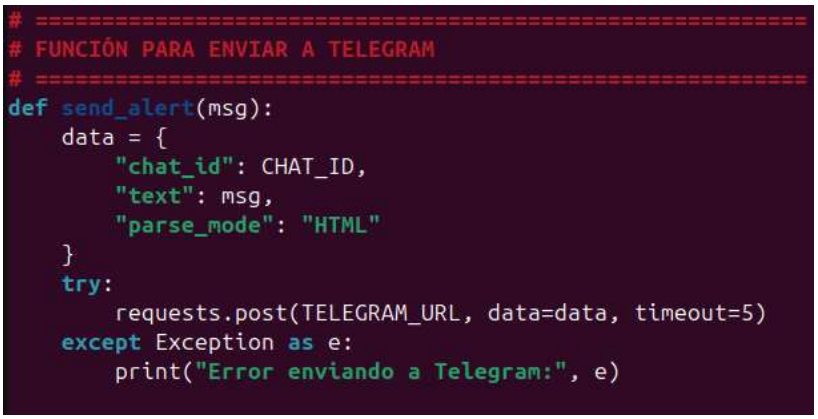


Figura 64 - Función de código para él envió de mensaje

Esto gestiona fallas de red, reconexiones y tiempo de espera.

7. Función para Formatear Mensajes

Esta función es el módulo que convierte una alerta cruda en un mensaje elegante y estructurado. Los datos de Suricata y Zeek llegan en formato JSON o texto desordenado.

Esta función los transforma en mensajes y hace que llegue al operador y registre el ataque, para poderlo seguir usando como se muestra en la siguiente *Figura 76*:

```
# ============================================================
# FORMATO DEL MENSAJE
# ============================================================
def build_message(sensor, ids_name, sev, signature, src, dst, proto, timestamp):
    fecha = timestamp.split("T")[0]
    hora = timestamp.split("T")[1].split(".")[0]

    return f"""
🌐🔎<b>INCIDENTE EN TRÁFICO DE RED - SERVER_IDS / LABORATORIO TEL UPSE</b> 🏴

🖥️<b>Sensor:</b> RACK 4
🛡️<b>IDS:</b> {ids_name}

📊 <b>Severidad:</b> {sev}
🔍 <b>Signature:</b> {signature}

📡 <b>Origen:</b> {src}
🎯 <b>Destino:</b> {dst}
📨 <b>Protocolo:</b> {proto}

📅 <b>Fecha:</b> {fecha}
⏰ <b>Hora:</b> {hora}

⚠️<b>Atención requerida.</b>
"""
```

Figura 65 – Formato y Función del mensaje

Para el proyecto funciona muy bien ya que proporciona un formato estándar, profesional y comprensible para el personal técnico

8. Procesador de alertas Suricata

Esta Función es un lector que recorre el eve. json línea por línea buscando únicamente eventos tipo alerta. Suricata registra miles de eventos. El script filtra solo:

- ataques
- anormalidades
- eventos críticos

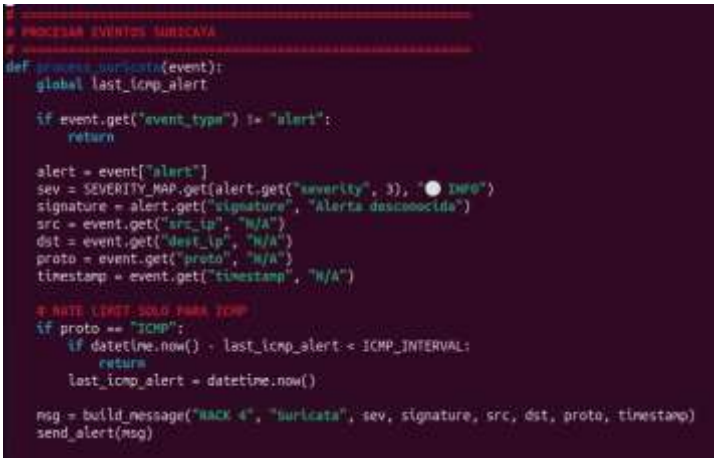


Figura 66 – Función para Procesar Eventos Suricata

Así como se muestra en la *Figura 77* que es de permitir generar alertas en tiempo real basadas en firmas.

9. Procesador de alteras Zeek

Esta función es un analizador del archivo notice.log que decodifica eventos de comportamiento y su apartado en el Script actúa como en la *Figura 78*. Zeek detecta patrones avanzados como:

- Escaneos
- Certificados vencidos
- Anomalías de tráfico
- Intentos de fuerza bruta

```python
# ============================================================
# PROCESAR EVENTOS ZEEK
# ============================================================
def process_zeek(line):
    try:
        fields = line.strip().split("\t")
        if len(fields) < 8:
            return

        timestamp = fields[0]
        src = fields[2]
        dst = fields[3]
        signature = fields[6]
        sev = SEVERITY_MAP.get(3, "🟡 MEDIA")  # Zeek no envía severidad real

        msg = build_message("RACK 4", "Zeek", sev, signature, src, dst, "N/A", timestamp)
        send_alert(msg)

    except Exception as e:
        print("Error procesando Zeek:", e)
```

Figura 67 – Procesador de Eventos de Zeek

Función dentro del proyecto es complementar a Suricata añadiendo detección basada en comportamiento, fortaleciendo el laboratorio.

10. Bucle principal del Sistema

Este bucle mostrado en la *Figura 79*, tiene como función garantizar a que el bot haga arrancar solo con el sistema, se reinicie si existe alguna falla y opere sin supervisión humana.

```python
# ============================================================
# BUCLE PRINCIPAL
# ============================================================
def main():
    print("BOT IDS Activo - Suricata + Zeek")

    # Abrir archivos
    f_suri = open(SURICATA_EVE, "r")
    f_suri.seek(0, 2)

    f_zeek = open(ZEEK_NOTICE, "r")
    f_zeek.seek(0, 2)

    while True:
        line_suri = f_suri.readline()
        if line_suri:
            try:
                event = json.loads(line_suri)
                process_suricata(event)
            except:
                pass

        line_zeek = f_zeek.readline()
        if line_zeek:
            process_zeek(line_zeek)

        time.sleep(0.2)

if __name__ == "__main__":
    main()
```

Figura 68 - Bucle Principal del Sistema Automatizado

11. Integración con systemd

El script se ejecuta como servicio, esto siendo parte de la integración de systemd para este script garantice que el bot:

- Arranque solo con el sistema
- Se reinicie si falla
- Opere sin supervisión humana

```
tesisgerv@tesisgerv:~$ sudo systemctl enable ids_alert_bot
tesisgerv@tesisgerv:~$ sudo systemctl start ids_alert_bot
```

Figura 69 – Actualización del estado del ids_alert_bot para su buena funcionalidad

Y es así como esta función convierte al script personalizado en un componente estable y profesional, digno de un entorno académico/laboratorio *Figura 80*.

12. Verificación del Estado

En la *Figura 81* podemos observar que la ejecución de estos parámetros, están en un estado Activo dando al script como una ejecución exitosa:

```
tesisgerv@tesisgerv:~$ sudo systemctl status ids_alert_bot
● ids_alert_bot.service - IDS Alert Bot for Suricata + Zeek
     Loaded: loaded (/etc/systemd/system/ids_alert_bot.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-12-05 04:48:58 -05; 5h 25min ago
   Main PID: 15986 (python3)
      Tasks: 1 (limit: 9432)
     Memory: 16.7M (peak: 17.4M)
        CPU: 22.557s
     CGroup: /system.slice/ids_alert_bot.service
             └─15986 /usr/bin/python3 /usr/local/bin/ids_alert_bot.py
```

Figura 70 - Estado Running de ids_alert_bot. service

CAPÍTULO IV

4.1 Resultados y Análisis del Sistema IDS Implementado

El presente capítulo expone los resultados alcanzados durante la implementación del sistema de monitoreo y detección de intrusos desarrollado para el Laboratorio de Telecomunicaciones. Estos resultados se encuentran directamente vinculados con el Objetivo General, que plantea diseñar e integrar una arquitectura IDS capaz de detectar, registrar y visualizar eventos de seguridad, así como con los Objetivos Específicos, que incluyen:

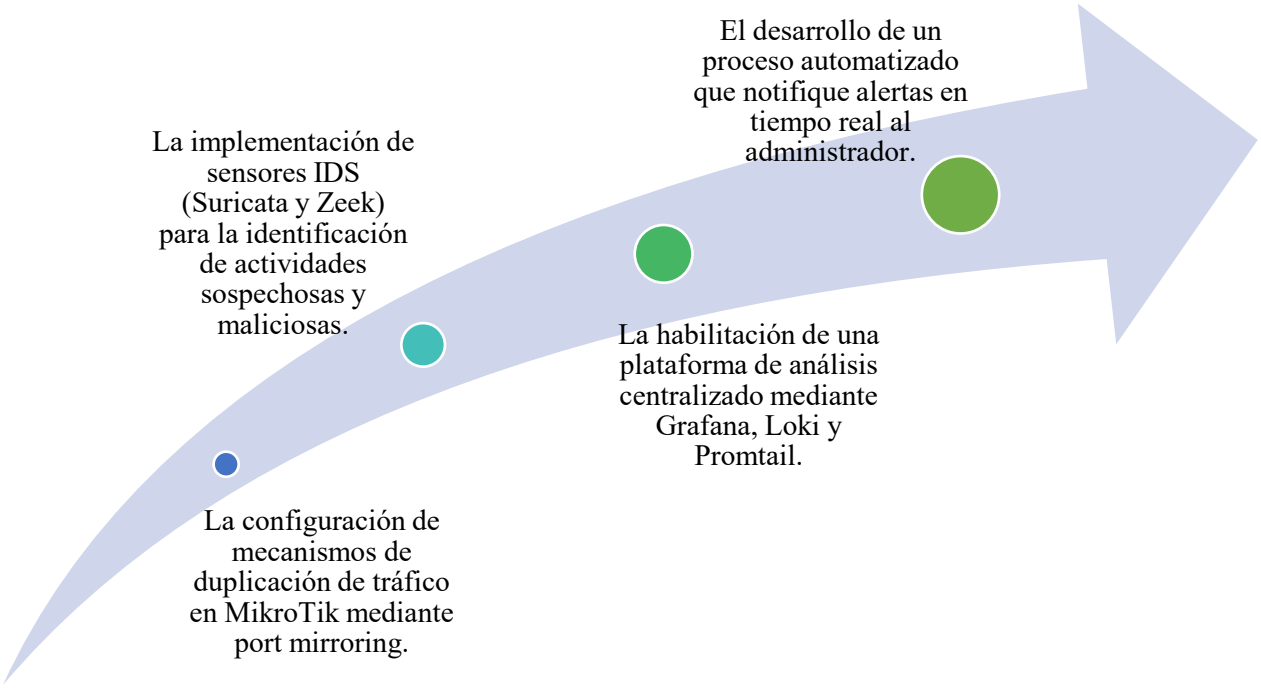


Ilustración 3 – Proceso de los objetivos propuestos cumplidos

Este capítulo presenta evidencias del funcionamiento de cada componente, detalla las pruebas realizadas con tráfico controlado y analiza el desempeño de los sensores dentro del entorno experimental. De esta manera, se documentan las visualizaciones generadas en el Dashboard de Grafana junto con la respuesta automatizada del bot de Telegram.

La secuencia de resultados reflejada a continuación permite evaluar la efectividad de la arquitectura, mostrar su coherencia con los objetivos planteados y destacar su aporte al fortalecimiento de la seguridad en infraestructuras educativas.

4.1.1 Resultados del Monitoreo en MikroTik mediante Port Mirroring

La primera etapa de la implementación consistió en habilitar un entorno de observación del tráfico dentro del enrutador MikroTik, configurando sus puertos como puntos de captura en tiempo real. Esta configuración permitió establecer una ruta clara para el flujo de información hacia los sensores IDS, acorde al primer objetivo específico, que señala la necesidad de disponer de un entorno seguro y controlado para el análisis del tráfico.

- **Configuración Aplicada al MikroTik CRS310/RB4011**

La configuración del port mirroring se realizó asignando un puerto origen para la recepción del tráfico y un puerto espejo para reenviar dicho tráfico hacia los servidores donde operaban Suricata y Zeek. Esto convirtió al MikroTik en un punto de observación central que permitió dirigir, sin intervención del usuario, cada paquete procesado hacia los sensores para su evaluación.

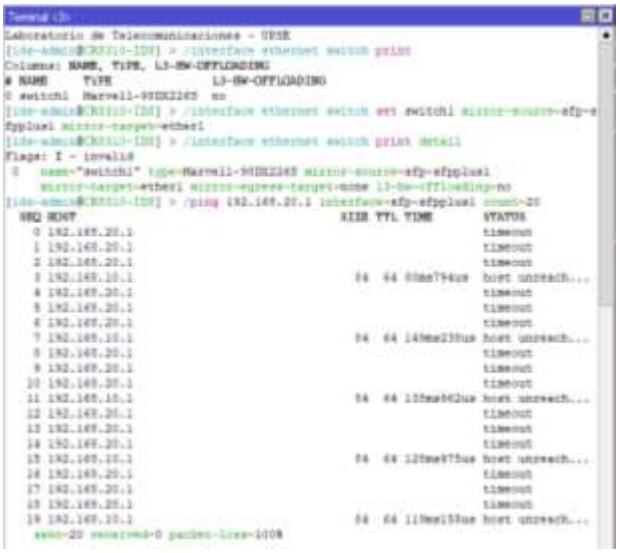


Figura 71 - Configuración del Port Mirroring en MikroTik

La *Figura 82* correspondiente muestra la sección del menú donde se estableció el puerto espejo, evidenciando la estructura interna del flujo replicado.

- ✓ **Verificación de la Duplicación de tráfico**

Una vez establecida la configuración, se verificó el funcionamiento del mirroring mediante herramientas nativas del MikroTik como Torch y el analizador de paquetes como se muestra en la *Figura 83*. Estas pruebas confirmaron que el tráfico de las interfaces seleccionadas estaba siendo replicado correctamente hacia los sensores IDS.

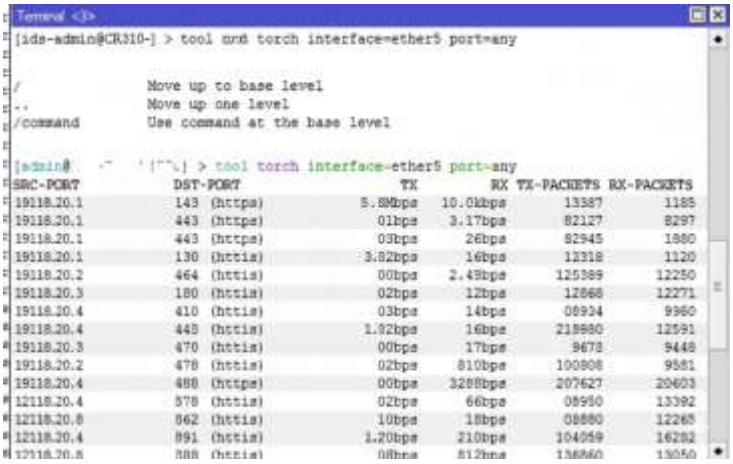


Figura 72 - Evidencia del tráfico observado mediante Torch.

- ✓ **Relevancia para el sistema IDS**

El port mirroring no constituye únicamente una acción de configuración ya que esto representa el puente que conecta la infraestructura física con la capa de análisis del sistema. Sin este proceso, ni los sensores ni el Dashboard tendrían información que procesar o visualizar. Por ello, su correcta operación se considera un componente clave para garantizar la continuidad de los procesos descritos en las siguientes secciones.

4.1.2 Resultados del Análisis de Tráfico en los Sensores IDS

Esta sección presenta los resultados obtenidos durante las pruebas de tráfico ejecutadas para evaluar el desempeño de los sensores IDS implementados: Suricata y Zeek. El análisis se enfoca en mostrar la capacidad del sistema para identificar patrones sospechosos, eventos anómalos y comportamientos potencialmente maliciosos dentro del entorno de red configurado. Esta etapa responde directamente al Objetivo General y al primer objetivo específico, demostrando la efectividad del sistema IDS en la detección y análisis en tiempo real.

- ✓ **Generación de Trafico de Pruebas**

Para evaluar el funcionamiento del sistema, se diseñaron tres escenarios de tráfico controlado:

1. **Tráfico Normal**

Se incluyó navegación básica, consultas DNS, pings y conexiones legítimas. Este escenario buscó comprobar que los sensores no generaran falsos positivos.

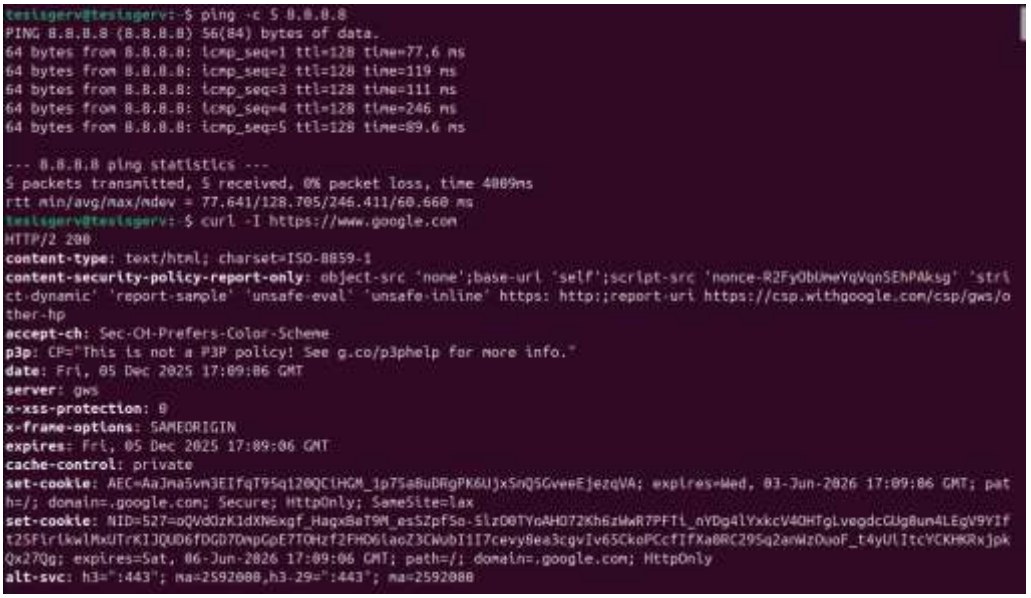


Figura 73 - Trafico Benigno Generado desde la Terminal de PC Server_IDS

2. Tráfico Sospechoso

Se ejecutaron paquetes intensivos ICMP, accesos reiterados a puertos no autorizados y acciones anómalas de exploración ligera.

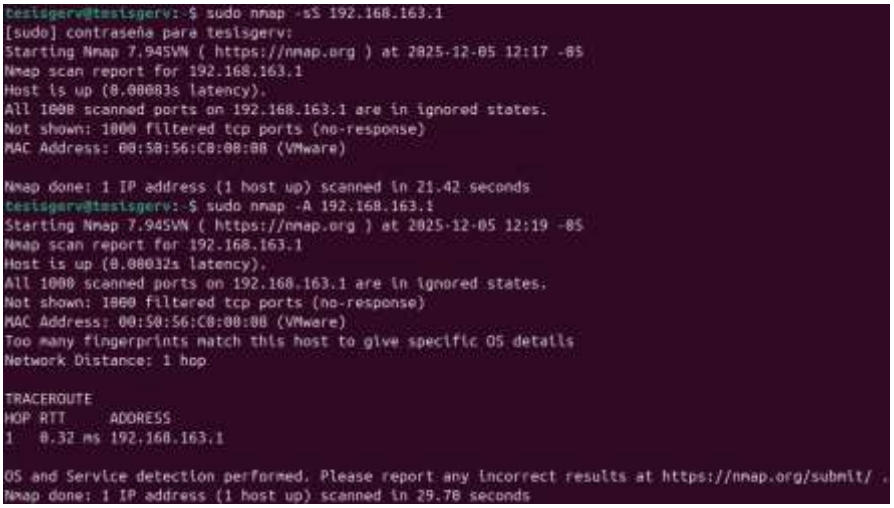


Figura 74 - Generando Trafico Sospecho dentro del Ambiente PC Server_IDS

3. Tráfico malicioso

Se simularon patrones similares a los utilizados por herramientas de reconocimiento como escaneos SYN y conexiones rápidas sucesivas, generando un entorno adecuado para validar la capacidad de alerta de ambos sensores.

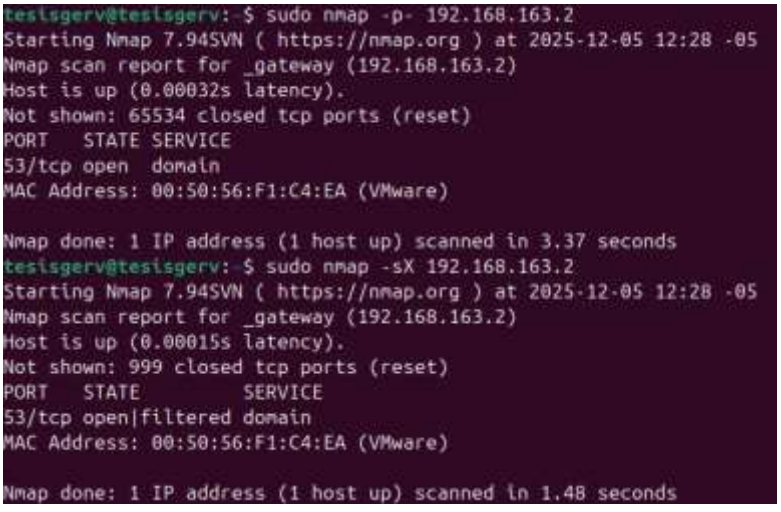


Figura 75 - Generando Trafico Malicioso dentro de la Terminal PC Server_IDS

Tabla 21 - Comandos utilizados para la generación de tráfico normal, sospechoso y malicioso en el entorno de pruebas IDS

TIPO DE TRÁFICO	COMANDO DE EJECUCIÓN	DESCRIPCIÓN
Normal	ping -c 5 192.168.163.2	Genera paquetes ICMP estándar hacia el gateway NAT, simulando actividad legítima de un usuario.
Normal	curl -I https://www.google.com	Solicitud HTTP legítima utilizada para simular navegación web habitual.
Sospechoso	sudo nmap -sS 192.168.163.2	Escaneo SYN emergente que indica reconocimiento inicial del sistema; actividad considerada anómala.
Sospechoso	sudo nmap -A 192.168.163.2	Escaneo agresivo con fingerprinting; comportamiento típico de auditorías no autorizadas.
Malicioso	sudo nmap -p- 192.168.163.2	Escaneo completo de puertos (full scan), característico de actividades ofensivas y mapeo de superficies de ataque.
Malicioso	sudo nmap -sX 192.168.163.2	XMAS scan que manipula flags TCP para evadir firewalls; considerada actividad maliciosa encubierta.

✓ **Detección realizada por Suricata**

Suricata registró eventos en su archivo fast.log, clasificando las alertas en función de su severidad. Durante las pruebas, este sensor mostró un desempeño eficiente, detectando:

- Intentos de escaneo de puertos
- Conexiones no autorizadas a servicios deshabilitados
- Actividad ICMP anómala
- Comportamientos que coincidían con firmas conocidas en su motor de reglas

Suricata se destacó por su análisis basado en firmas, lo que permitió identificar patrones claros y específicos de ataque mostrados en la *Figura 87*.

Figura 76 - Fragmento del archivo fast.log mostrando una alerta detectada por Suricata.

✓ Detección realizada por Zeek

Zeek aportó un análisis distinto y su enfoque se centra en describir eventos, comportamientos y relaciones entre conexiones. Durante las pruebas produjo archivos como:

- conn.log = Registro completo de conexiones
- notice.log = Actividades inusuales
- weird.log = Eventos fuera de los estándares esperados

Figura 77 - Eventos registrados por Zeek en conn.log

Este sensor permitió observar detalles como:

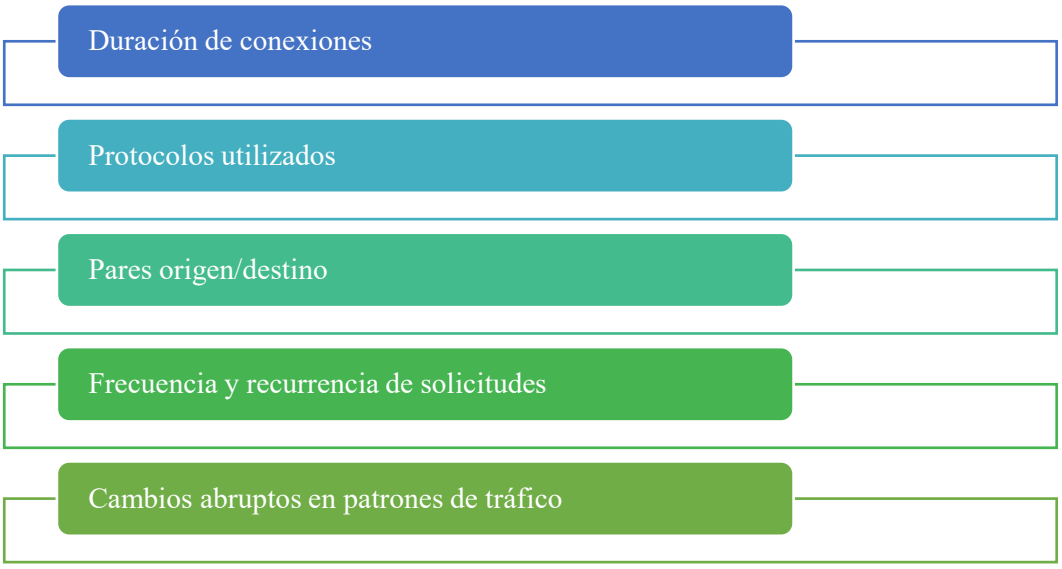


Ilustración 4 - Detalles Observados por el Sensor Zeek

Zeek enriqueció el análisis al ofrecer una visión contextual de cada interacción.

- ✓ **Relación entre el tráfico generado y la respuesta de los sensores**

Los resultados muestran que los dos sensores actuaron de manera complementaria:

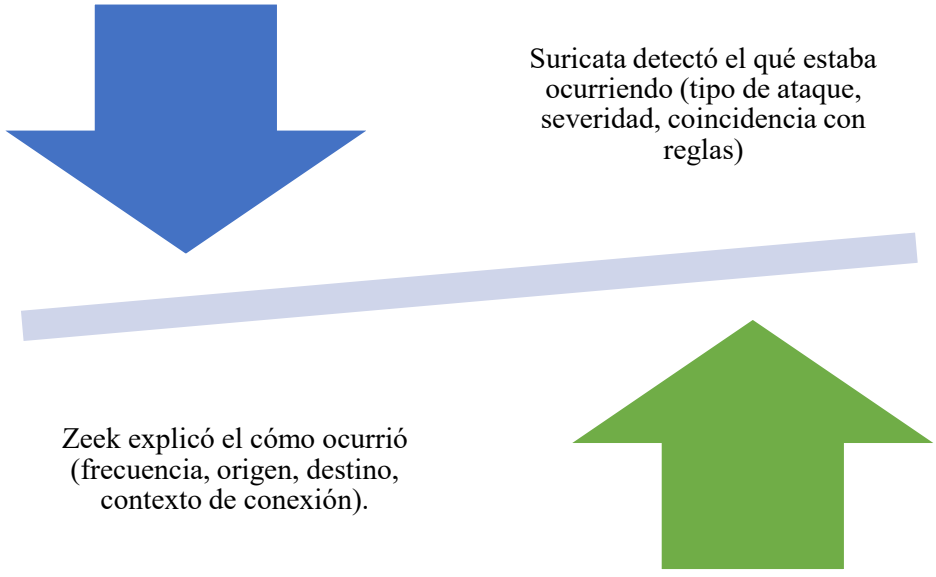


Ilustración 5 - Relación entre Sensores (Suricata/Zeek)

Esta combinación permitió obtener una visión más completa del comportamiento de la red, fortaleciendo el análisis del sistema IDS como se muestra en la *Tabla 22*.

Tabla 22 - Resumen del Trafico Generado y Respuestas de los sensores

ESCENARIO	TIPO DE TRÁFICO	SURICATA	ZEEK	NIVEL DE RIESGO
Normal	Navegación / DNS	No alertas	Logs informativos	Bajo
Sospechoso	ICMP intensivo / Accesos reiterados	Media / Baja	Notices relevantes	Medio
Malicioso	Escaneo / Conexiones rápidas	Alta	Notice + Weird	Alto

4.1.3 Procesamiento y Almacenamiento de Logs con Loki y Promtail

El procesamiento de los eventos detectados por los sensores IDS se realizó mediante la integración de Promtail como agente de recopilación y Loki como base de almacenamiento de logs. Esta etapa permitió centralizar la información generada por Suricata y Zeek, dando cumplimiento al segundo objetivo específico, que planteaba la necesidad de contar con una plataforma de visualización capaz de monitorear alertas y métricas en tiempo real.

El uso de esta arquitectura permitió transformar registros independientes y dispersos en un flujo estructurado y accesible desde Grafana, facilitando el análisis posterior y permitiendo una interpretación más clara sobre los patrones de tráfico dentro del entorno monitoreado.

- ✓ **Recolección de logs a través de Promtail**

Promtail fue configurado para monitorear de manera continua los archivos generados por ambos sensores. El agente identificó nuevas entradas desde que se generaron tráfico tanto Normal, Sospechoso y Maligno las cuales son las que se muestran en la *Figura 89.*

```
tesisgerv@tesisgerv:~$ sudo cat /var/lib/promtail/positions.yaml
[sudo] contraseña para tesisgerv:
positions:
  /opt/zeek/logs/current/conn.log: "14423"
  /opt/zeek/logs/current/dns.log: "8334"
  /opt/zeek/logs/current/ssl.log: "1081"
  /opt/zeek/logs/current/stats.log: "469"
  /opt/zeek/logs/current/stderr.log: "22"
  /opt/zeek/logs/current/stdout.log: "204"
  /opt/zeek/logs/current/telemetry.log: "167973"
  /opt/zeek/logs/current/weird.log: "1554"
  /var/log/suricata/eve.json: "178006446"
  /var/log/suricata/fast.log: "531419"
  /var/log/suricata/stats.log: "0"
  /var/log/suricata/suricata.log: "120515"
```

Figura 78 - Configuración del archivo Promtail. yaml mostrando las rutas monitoreadas.

- ✓ **Almacenamiento e indexación de eventos en Loki**

Loki recibió y almacenó los logs utilizando un modelo orientado a etiquetas, evitando la necesidad de indexar el contenido completo como se muestra en la *Figura 90*. Esto permitió:

- Consultas rápidas sobre miles de líneas de logs.
- Filtrado por sensor, nivel de severidad o patrón específico.
- Visualizaciones generadas de forma eficiente en Grafana.

El sistema demostró ser adecuado para el volumen de registros generado durante las pruebas, preservando la estructura cronológica de los eventos y facilitando su correlación.

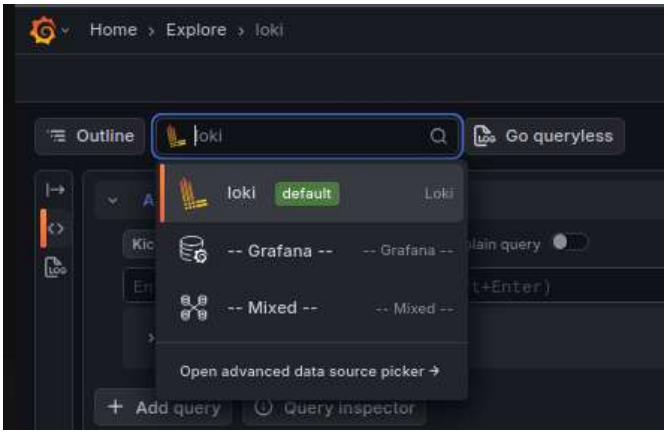


Figura 79 - Panel de consulta en Loki para mostrar la indexación de eventos IDS

✓ **Consulta y validación de logs desde Grafana**

Una vez almacenados en Loki, los eventos fueron consultados mediante el panel integrado en Grafana. Desde esta interfaz fue posible verificar alertas recientes, identificar patrones de actividad en intervalos de tiempo específicos. Comparar la frecuencia de eventos entre los sensores y analizar comportamientos anómalos mediante consultas personalizadas.

Las consultas revelaron coincidencias claras con los escenarios de prueba ejecutados. Por como se muestra en las *Figuras 91 y 92*, los escaneos generaron aumentos visibles en las notificaciones de Suricata, mientras que Zeek registró conexiones inusuales de corta duración.

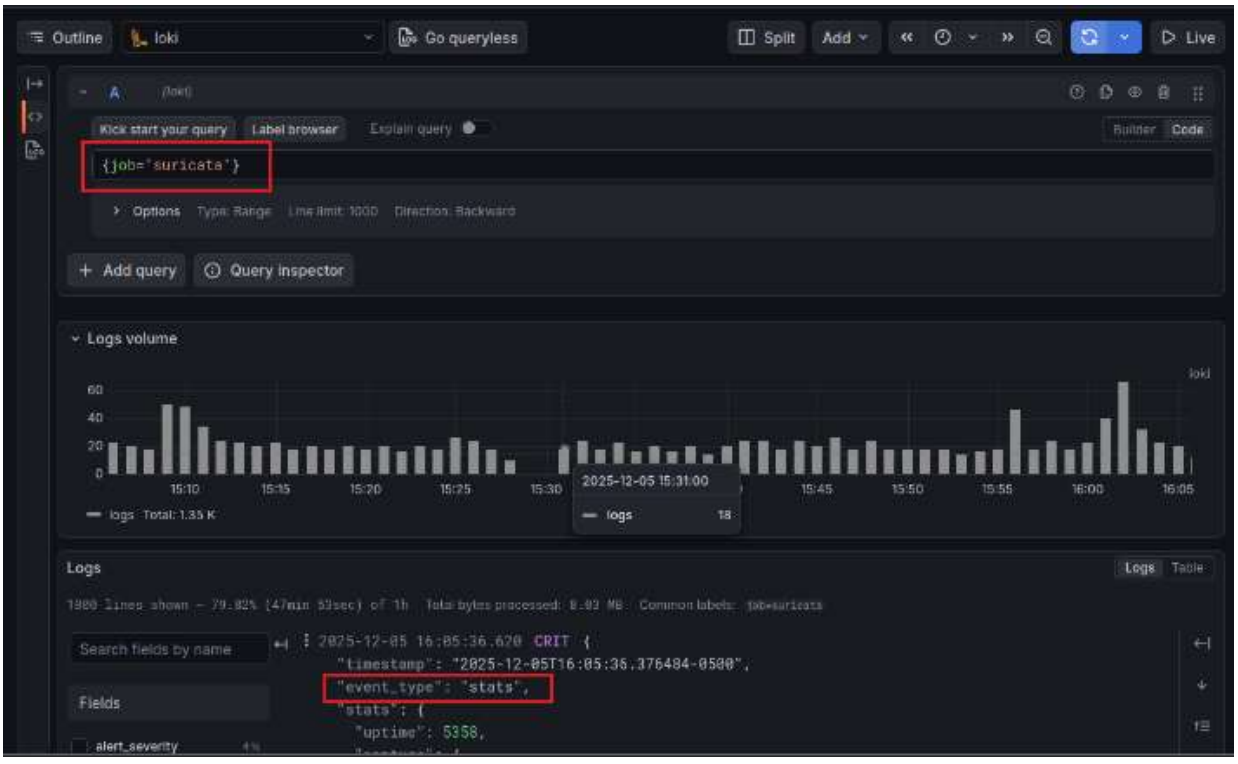


Figura 80 - Consulta en Grafana mostrando logs filtrados por Suricata.

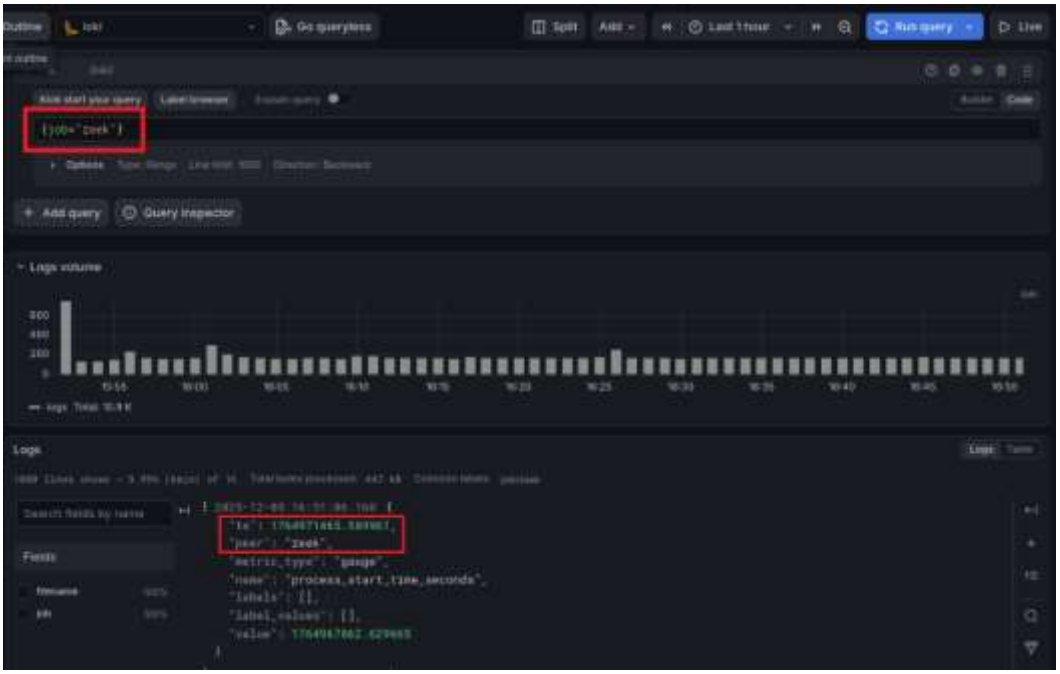


Figura 81 - Consulta en Grafana mostrando eventos generados por Zeek.

4.1.4 Visualización del Comportamiento del Sistema IDS mediante el Dashboard en Grafana

La construcción del Dashboard en Grafana permitió consolidar los eventos generados por Suricata y Zeek en una plataforma gráfica de visualización en tiempo real. Esta etapa responde de manera directa al segundo objetivo específico, orientado al diseño e implementación de un entorno que facilite el monitoreo continuo del sistema IDS en el Laboratorio de Telecomunicaciones.

A través de los paneles diseñados, fue posible analizar patrones del tráfico, identificar eventos con distintos niveles de severidad y correlacionar el comportamiento registrado por los sensores con el tráfico generado durante las pruebas. El Dashboard se convirtió así en el elemento central de interpretación, permitiendo transformar información técnica en observaciones comprensibles y accionables.

- ✓ **Paneles Funcionales IDS_LAB_TELECO_UPSE Dashboard Grafana**

El Dashboard se diseñó con una estructura organizada en secciones para facilitar la lectura e interpretación del estado de la red. Su composición incluyó tres filas para distribuirlas de manera como Fila 1 Actividades Globales, Fila 2 Análisis de Protocolo y Severidad Y Fila 3 Amenazas Especificas

Fila 1 – Actividad Global del IDS / Panorama General

- **Panel 1 - Comparación de Actividad IDS: Suricata vs Zeek (Eventos por Minuto)**

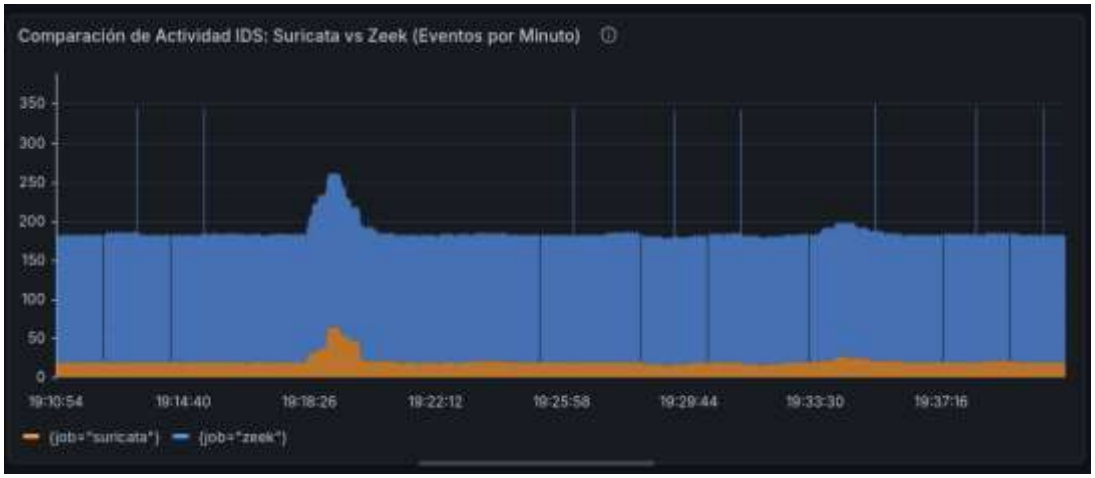


Figura 82 – Panel 1 Comportamiento de los Sensores IDS en el Ambiente del Laboratorio

Este panel 1 *Figura 93* permite comparar ambos motores IDS. Suricata genera alertas basadas en reglas, mientras Zeek registra eventos de red contextualizados en el intervalo de tiempo de 1 minuto para llevar su comportamiento seguido. La relación entre ambos evidencia la carga de análisis en tiempo real, grafica creada y ejecutada por los JSON de la *Figura 62*.

- **Panel 2 - Tendencia Temporal de Alertas por Severidad (Suricata)**

El panel evidencia los momentos en los que se generó mayor actividad maliciosa, permitiendo correlacionar ataques con eventos específicos en la red, la creación de este panel es del código generado en el Explore de Loki de la *Figura 63*.

Figura 83 – Panel de Tendencia Temporal de Alertas por Severidad (Suricata)

Fila 2 – Clasificación de Alertas

- **Panel 3 – Distribución de Alertas por Protocolo (Suricata)**

Este panel es creado por la ejecución del código en JSON que se ejecutó en la *Figura 64* y nos muestra qué protocolos fueron más vulnerados durante las pruebas. Es típico ver actividad destacada en TCP debido a scans y conexiones anómalas. Y su interpretación es:

Figura 84 – Panel de Distribución de Alertas por Protocolo (Suricata)

- **Panel 4 – Clasificación de Alertas por Severidad (Suricata)**

Este panel permite distinguir la criticidad de los eventos detectados, clave para priorizar planes de respuesta ante incidentes, esta grafica fue creada con código de ejecución tipo JSON que se encuentra en la *Figura 63*.

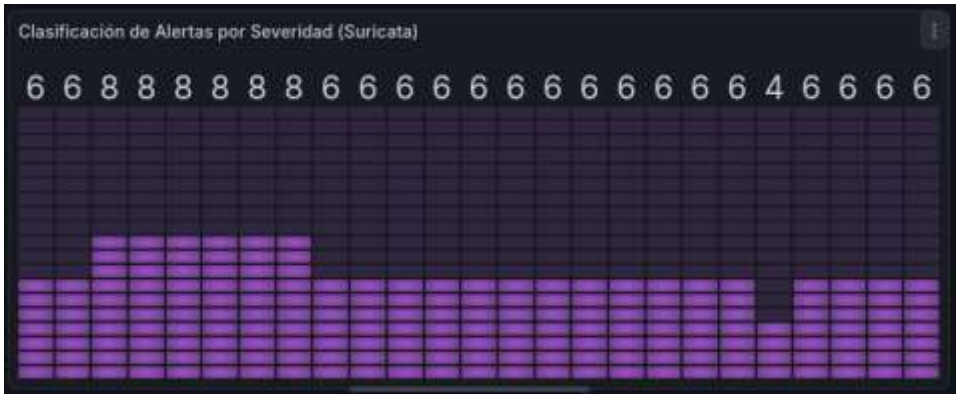


Figura 85 - Panel Clasificación de Alertas por Severidad (Suricata)

- **Panel 5 – Conteo Total de Alertas IDS (Suricata)**

Este contador permite validar la correlación entre actividad de red y notificaciones enviadas al bot de Telegram.

Figura 86 – Panel de conteo total de Alertas IDS por el Sensor (Suricata)

Fila 3 – Inteligencia de Amenazas

- **Panel 6 - Top 5 IPs con Mayor Número de Alertas (Suricata)**

Identifica los hosts que originan más incidentes. Útil para correlacionar pruebas ofensivas de la MV y detectar posibles anomalías, este panel se creó ejecutando el código JSON en Loki la cual está en la *Figura* .

Figura 87 – Panel Top 5 IPs con Mayor Numero de Alertas

4.1.5 Resultados del Sistema de Notificación Automatizado mediante Telegram

El sistema de notificación implementado permitió validar la capacidad del IDS para alertar en tiempo real cuando Suricata o Zeek registraron eventos relevantes. Esta sección presenta exclusivamente los resultados obtenidos, mostrando ejemplos reales de mensajes recibidos, el tipo de alertas generadas y la utilidad práctica de este mecanismo dentro del entorno de monitoreo.

- ✓ **Alertas generadas por Suricata según su Severidad.**

Durante las pruebas de tráfico malicioso, Suricata emitió alertas clasificadas como Alta, Media y Baja severidad. Cada alerta fue captada por el script y enviada inmediatamente a Telegram.

Los mensajes recibidos presentaron la siguiente estructura:

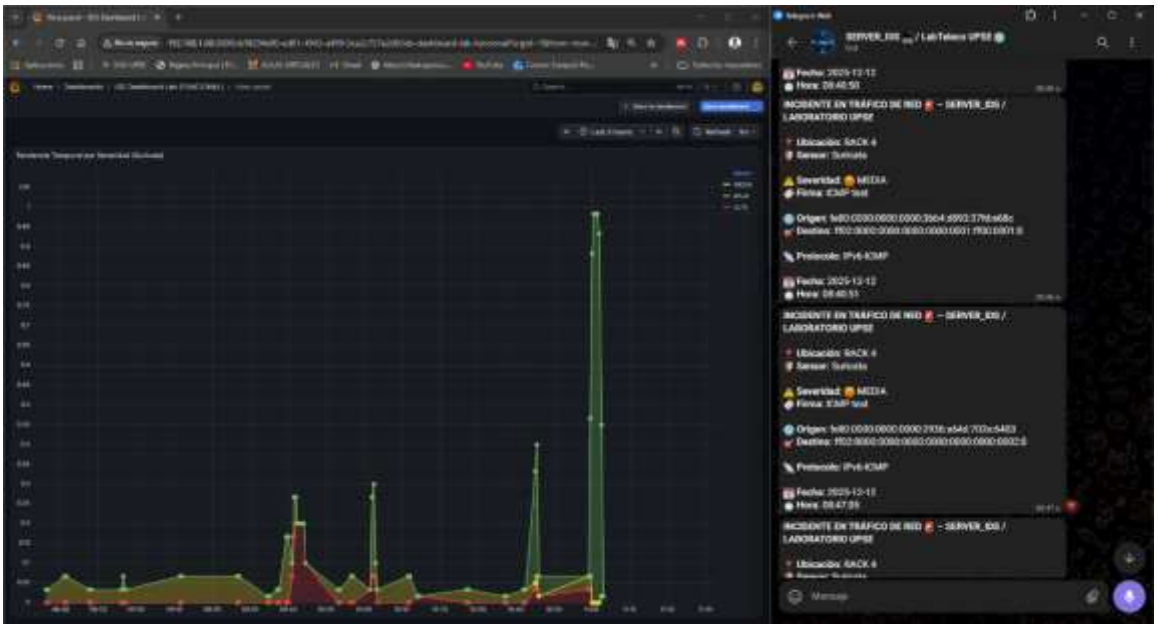


Figura 88 - Comportamiento en Grafana y verificación con Telegram

La *Figura 98* muestra cómo las alertas generadas por Suricata varían a lo largo del tiempo según su nivel de severidad. En los primeros minutos se observa un comportamiento estable con

eventos de baja intensidad, propios del tráfico normal. A medida que se ejecutaron las pruebas de ICMP y los escaneos de puertos, la severidad aumentó y aparecieron picos más visibles, especialmente en la categoría Media. Este patrón confirma que el sensor respondió correctamente ante actividades anómalas, registrando de forma clara cada incremento de riesgo durante las pruebas realizadas.

- **Severidad Baja – Escenario ICMP**

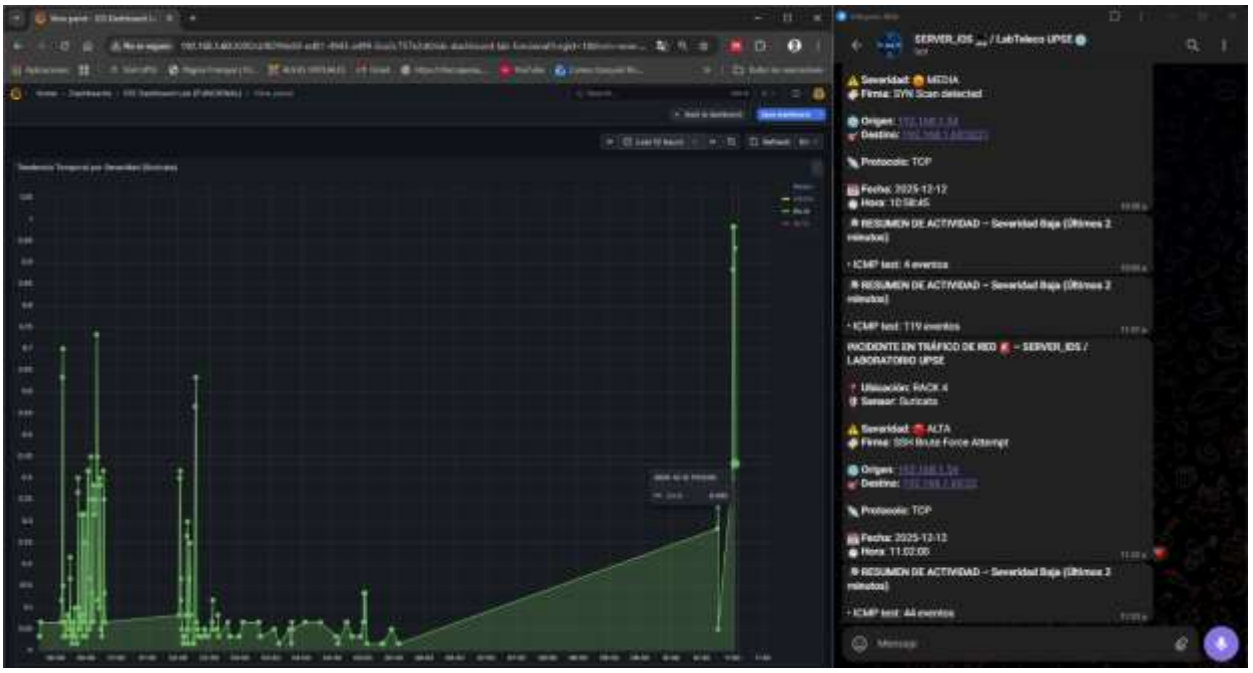


Figura 89 - Comportamiento de Severidad Baja – Suricata y Notificación en Telegram

La *Figura 99* evidencia cómo la severidad de las alertas generadas por Suricata fue aumentando conforme se aplicaron pruebas más agresivas. Durante la madrugada las fluctuaciones se mantuvieron en niveles bajos lo que coincide con periodos de tráfico benigno. A partir de las 09:30 se observan los primeros incrementos esto reflejando los ICMP intensivos que el bot de Telegram reportó como “ICMP test”.

Minutos después la severidad asciende a valores medios y altos. Estos picos coinciden de forma directa con las notificaciones de “SYN Scan detected” y “SSH Brute Force Attempt”, confirmando que el sensor logró identificar correctamente los cambios bruscos en el comportamiento del tráfico. El punto más elevado del gráfico cercano al valor 1.0 aparece al mismo tiempo que Telegram reporta un ataque de fuerza bruta, demostrando una sincronización clara entre la visualización en Grafana y las alertas automáticas enviadas al administrador.

Esto en conjunto la gráfica y los mensajes recibidos muestran un sistema coherente: cada actividad sospechosa se refleja primero en el panel y luego se confirma mediante la notificación inmediata en Telegram, lo que valida el funcionamiento integral del IDS dentro del laboratorio.

Tabla 23 - Tabla cuantitativa de la Grafica "Tendencia temporal por severidad (Suricata)"

Hora aproximada	Severidad	Valor (escala relativa)	Tipo de evento asociado	Coincidencia con Telegram
00:10 – 00:30	Baja	0.40 – 0.75	ICMP repetitivo / tráfico ligero	No genera alertas (actividad normal)
09:30 – 09:40	Media	0.15 – 0.25	ICMP intensivo	Sí: *"ICMP test"*
09:45 – 09:50	Alta	0.18 – 0.22	Escaneo SYN y XMAS	Sí: *"SYN Scan detected"*
10:00 – 10:05	Media	0.23	Reconocimiento con TCP	Sí: Notificación de severidad media
10:50 – 11:00	Media/Alta	0.90 – 1.00	Ataques dirigidos (SSH brute force)	Sí: *"SSH Brute Force Attempt"*
11:02 – 11:03	Media	0.05 – 0.08	Nuevo ciclo de ICMP	Sí: resumen de actividad

En la *Tabla 23* los valores numéricos corresponden a la escala relativa que Grafana muestra para la intensidad de cada evento. No representan bytes ni paquetes sino la ponderación visual de la severidad.

- **Severidad Media - Escenario SYN Scan**

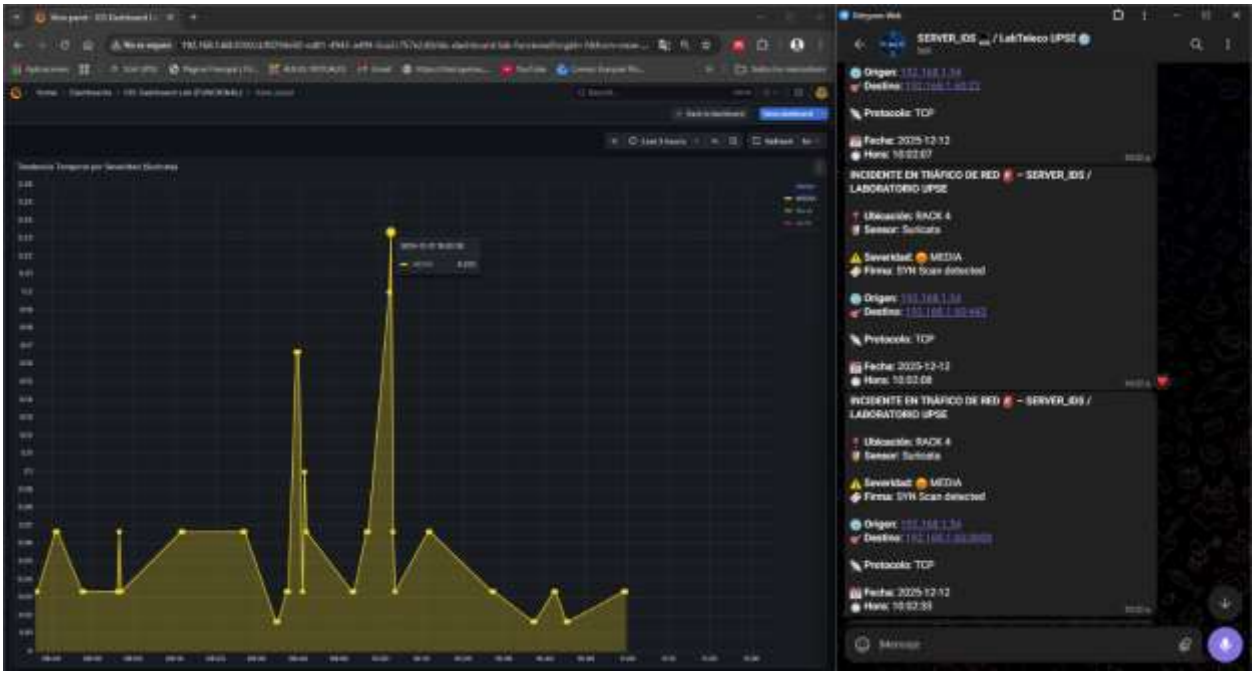


Figura 90 - Comportamiento de Severidad Media – Suricata y Notificación en Telegram

La *Figura 100* refleja un comportamiento más concentrado en la severidad Media propio de los escaneos SYN realizados durante la prueba. En los primeros minutos los valores se mantienen bajos y estables lo que corresponde al tráfico normal del entorno. Sin embargo a partir de las 09:40 se observa un aumento progresivo que coincide con el inicio del reconocimiento TCP.

El punto más alto registrado alrededor de las 10:02 y coincide exactamente con las alertas que Telegram reporta como “SYN Scan detected” las cuales fueron enviadas en intervalos muy cercanos entre sí. Este comportamiento muestra que el sensor logró identificar la repetición de intentos de conexión orientados a distintos puertos, característico de un escaneo.

Después del pico mostrado la actividad disminuye gradualmente lo que esto indica que el ataque dejó de ejecutarse y el tráfico volvió a la normalidad. Esta correlación entre la curva de severidad

y los mensajes del bot confirma la correcta respuesta del IDS ante un escenario de reconocimiento activo.

Tabla 24 - Tabla Cuantitativa – Tendencia Temporal por Severidad (Suricata) – Escenario SYN Scan

Hora aproximada	Severidad	Valor (relativo en gráfica)	Evento observado	Coincidencia con Telegram
09:00 – 09:10	Media	0.05 – 0.08	Tráfico leve, fluctuación normal	No
09:20 – 09:30	Media	0.10 – 0.12	Comportamiento irregular previo a pruebas	No
09:40 – 09:50	Media	0.18 – 0.25	Inicio de reconocimiento TCP	Sí – SYN Scan detectado
10:02 – 10:03	Media	0.23 – 0.24	Escaneo SYN directo a diferentes puertos	Sí – Tres alertas consecutivas
10:10 – 10:20	Media	0.08 – 0.12	Actividad residual	No
10:30 – 10:40	Media	0.06 – 0.09	Tráfico estabilizado	No

Esta *Tabla 24* se basa en los puntos marcados en la gráfica y los tiempos registrados por el bot en Telegram.

- **Severidad Alta – Escenario SSH Brute Force Attempt**

En la *Figura 101* se aprecia un comportamiento claro que nos indica que la severidad Alta aparece de forma marcada durante el intervalo en el que se ejecutó el ataque de fuerza bruta sobre el servicio SSH. Los picos más notorios se concentran alrededor de las 08:47 a 08:50 exactamente cuándo Telegram reporta la alerta “SSH Brute Force Attempt” lo que confirma que el sensor reaccionó de manera inmediata frente a este tipo de agresión.

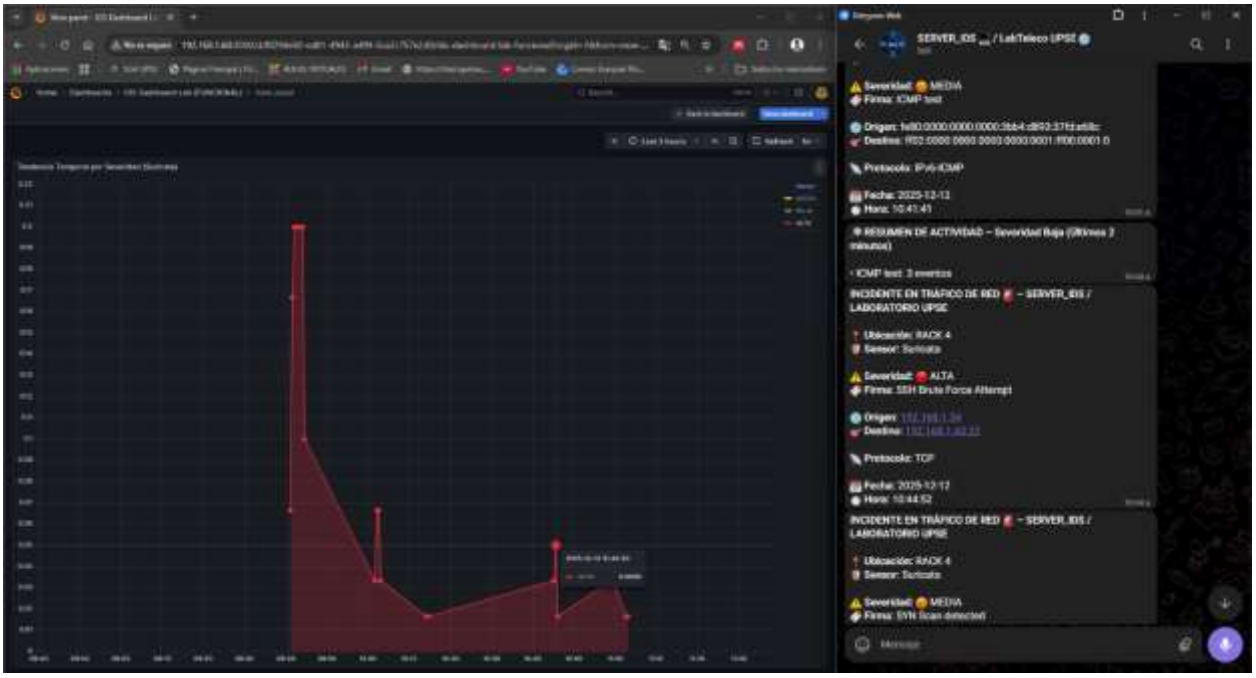


Figura 91 - Comportamiento de Severidad Alta – Suricata y Notificación en Telegram

Después del pico principal la curva desciende pero mantiene algunos valores aislados que reflejan intentos adicionales o conexiones anómalas posteriores. Más adelante cerca de las 10:40 se observa otro pequeño incremento coincidente con un SYN Scan, evidenciado también en los mensajes enviados por el bot.

En conjunto la gráfica y las alertas muestran que Suricata identificó correctamente los momentos críticos, destacando los ataques dirigidos contra servicios específicos y diferenciándolos de la actividad normal del entorno. Esto demuestra que la clasificación de severidad se ajusta de manera precisa al tipo de tráfico detectado.

4.2 Discusión de Resultados

Los resultados demuestran que la arquitectura IDS implementada logra detectar eficazmente actividades sospechosas y maliciosas dentro del laboratorio. El análisis combinado de Suricata y Zeek permitió capturar tanto ataques basados en firmas como comportamientos anómalos, confirmando la importancia de utilizar sensores complementarios.

La visualización en Grafana aportó un valor significativo al proceso de análisis ya que permitió identificar patrones temporales claros, picos de actividad y correlaciones entre severidades lo que coincide con las alertas recibidas a través de Telegram. La coordinación entre detección, visualización y notificación evidencia un sistema coherente y funcional.

A pesar de los resultados positivos, se identificaron limitaciones:

- ✓ El sistema depende del rendimiento del servidor local para procesar grandes volúmenes de tráfico.
- ✓ Zeek omitió ciertos eventos rápidos, mostrando una sensibilidad menor en escenarios de alta frecuencia.
- ✓ Suricata generó falsos positivos en contextos benignos, lo que podría generar ruido operativo.

Como mejoras se sugiere ampliar las reglas de Suricata, ajustar los umbrales de Zeek y considerar un sistema de correlación de eventos para reducir el impacto de FP/FN.

En términos de impacto la solución implementada fortalece la seguridad del laboratorio y ofrece una base sólida para futuras ampliaciones esto demostrando que es viable implementar un IDS profesional utilizando herramientas abiertas y un entorno de laboratorio académico.

4.3 Cumplimiento de Objetivos Específicos

- Objetivo 1: Configurar MikroTik para monitoreo IDS

  Se replicó tráfico en tiempo real hacia ambos sensores. Se evidenciaron capturas correctas en Torch y en los logs analizados.

- Objetivo 2: Implementar Dashboard en Grafana

  Se diseñaron paneles funcionales que muestran comparaciones, tendencias temporales, severidades, orígenes y protocolos.

- Objetivo 3: Sistema automatizado de alertas en Telegram

  El bot notificó con precisión eventos de severidad Alta, Media y Baja tal como se evidencia en las demostraciones.

CONCLUSIONES

En este trabajo se diseñó e implementó un sistema de detección de intrusos (IDS) capaz de identificar, registrar y notificar eventos de seguridad dentro del entorno del laboratorio de telecomunicaciones. El cumplimiento de los objetivos específicos permitió establecer un sistema de monitoreo funcional, que se valida con los siguientes puntos clave:

1. Se logró configurar adecuadamente los puertos del enrutador MikroTik para habilitar el port mirroring, lo que permitió dirigir el tráfico hacia los sensores IDS (Suricata y Zeek). Este paso fue fundamental para establecer el entorno seguro de monitoreo, cumpliendo el primer objetivo específico.

2. El desarrollo e implementación de un Dashboard en Grafana cumplió con su propósito de ofrecer una visualización clara y efectiva del tráfico en tiempo real, la severidad de las alertas y la distribución de los eventos. Los paneles creados mostraron la capacidad del sistema para identificar patrones y comportamientos anómalos en la red.

4. El sistema de alertas automatizado mediante Telegram cumplió su función de notificar en tiempo real sobre eventos críticos, como intentos de acceso no autorizado o escaneos de puertos. Este sistema mejoró la respuesta ante incidentes, permitiendo una intervención inmediata por parte del personal encargado.

RECOMENDACIONES

A pesar de la efectividad demostrada por Suricata y Zeek se recomienda optimizar las reglas de Suricata para reducir los falsos positivos (FP) especialmente en escenarios de tráfico benigno. Para Zeek se sugiere ajustar los umbrales de detección y mejorar su capacidad de identificar conexiones rápidas que actualmente se pasan por alto.

- Implementación de un Sistema de Correlación de Eventos SIEM

Se recomienda integrar un Sistema de Correlación de Eventos (SIEM) que permita analizar la información generada por los sensores de manera más eficiente. Esta implementación mejoraría la capacidad del sistema para correlacionar eventos de diferentes fuentes (Suricata, Zeek, logs de MikroTik), proporcionando una visión más completa de la seguridad de la red.

- Implementación de IPS

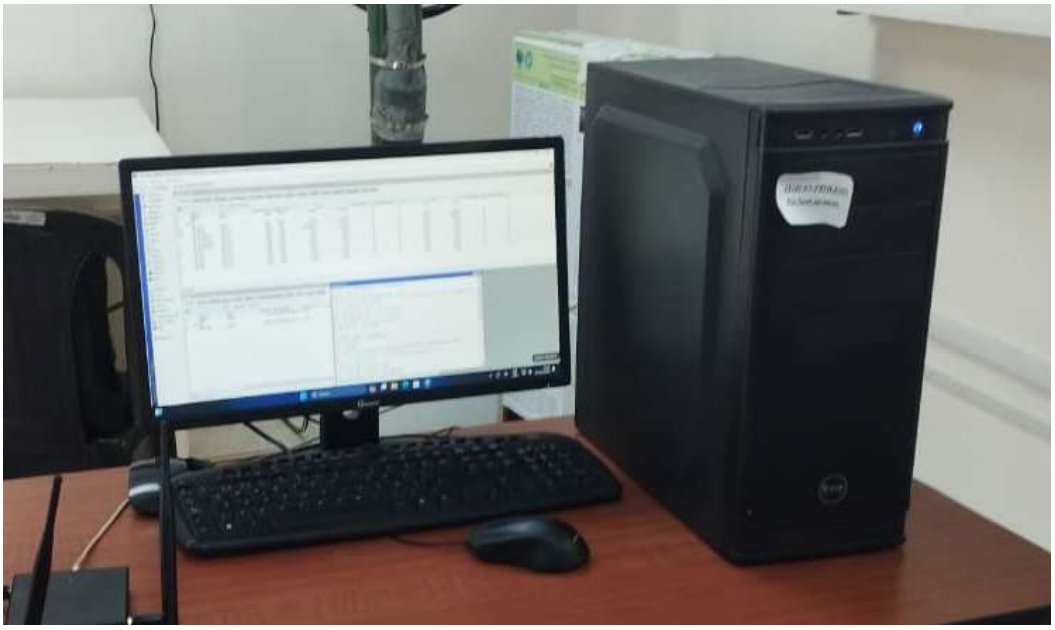
Como paso futuro se sugiere la implementación de un Sistema de Prevención de Intrusos (IPS) que, además de detectar eventos maliciosos como los IDS, pueda bloquear automáticamente el tráfico sospechoso en tiempo real. Esto aumentaría significativamente la seguridad al evitar que los ataques lleguen a afectar la red.

- Mejoras en el Desempeño del Sistema

Con el aumento de la carga de tráfico en redes más grandes, el rendimiento de los sensores podría verse afectado. Se recomienda realizar pruebas de estrés para garantizar que el sistema sea escalable y pueda manejar altos volúmenes de tráfico sin comprometer su eficiencia.

ANEXOS

Anexo A. Evidencias de Implementación en el Laboratorio



– Fotografías del montaje físico realizado

– Conexiones del MikroTik, servidor IDS y equipos de prueba

– Puesta en marcha del entorno de monitoreo

Anexo B. Comportamiento de los Sensores IDS (Suricata y Zeek)

– Capturas del tráfico detectado

– Fragmentos de logs generados durante las pruebas

– Registros de alertas clasificadas por severidad

Anexo C. Notificaciones Automáticas Enviadas por Telegram

– Alertas generadas por Suricata

– Eventos reportados por Zeek

– Ejemplos de notificaciones de severidad Alta, Media y Baja

– Evidencia del funcionamiento en tiempo real

BIBLIOGRAFÍAS

[1] A. K. B. Arnob, R. R. Chowdhury, N. A. Chaiti, S. Saha, y A. Roy, "A comprehensive systematic review of intrusion detection systems: emerging techniques, challenges, and future research directions", *Journal of Edge Computing*, vol. 4, no 1, pp. 73–104, may 2025, doi: 10.55056/jec.885.

[2] A. J. A. Immastephy y K. Punitha, "A Systematic Review on Network Intrusion Detection System based on machine learning and deep learning approach", en *E3S Web of Conferences*, EDP Sciences, jun. 2024. doi: 10.1051/e3sconf/202454014006.

[3] MD Shadman Soumik, "A comparative analysis of Network Intrusion Detection (NID) using Artificial Intelligence techniques for increase network security", *International Journal of Science and Research Archive*, vol. 13, no 2, pp. 4014–4025, dic. 2024, doi: 10.30574/ijsra.2024.13.2.2664.

[4] M. Janati y F. Messaoudi, "Intrusion Detection System-Based Network Behavior Analysis: A Systemic Literature Review". [En línea]. Disponible en: www.ijacsa.thesai.org

[5] | Webinar, "Building advanced Grafana dashboards".

[6] M. L. Septipalan, I. B. K. Widiartha, A. Zubaidi, y M. Taufik, "Integrated Notification System for Smart Parking Security Using Bot Telegram", *Jurnal Penelitian Pendidikan IPA*, vol. 10, no 5, pp. 2679–2686, may 2024, doi: 10.29303/jppipa.v10i5.7447.

[7] R. Man y R. A. Dissertation, "Intrusion Detection System for Internet of Things (IDS for IoT)", 2023.

[8] A. Ugbari y S. O. Adebayo, “A Comprehensive Review of Network Intrusion Detection Systems”, 2025. [En línea]. Disponible en: www.ijres.org

[9] Sridevi Kakolu, Muhammad Ashraf Faheem, y Muhammad Aslam, “AI-enabled intrusion detection systems in IoT networks: Advancing defense mechanisms for resource-constrained devices”, *International Journal of Science and Research Archive*, vol. 9, no 1, pp. 752–769, jun. 2023, doi: 10.30574/ijsra.2023.9.1.0316.

[10] M. Sajid *et al.*, “Enhancing intrusion detection: a hybrid machine and deep learning approach”, *Journal of Cloud Computing*, vol. 13, no 1, dic. 2024, doi: 10.1186/s13677-024-00685-x.

[11] M. B. Al-Doori y K. M. Ali Alheeti, “AI-Driven Features for Intrusion Detection and Prevention Using Random Forest”, *Journal of Cybersecurity and Information Management*, vol. 16, no 1, pp. 01–14, 2025, doi: 10.54216/JCIM.160101.

[12] “Encuesta sobre sistemas de detección de intrusiones basados en host y red | Solicitar PDF”. Accedido: 7 de diciembre de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/271208490_Survey_on_Host_and_Network_Based_Intrusion_Detection_System

[13] “145330024859988-service-definition-document-2024-05-07-1101”.

[14] I. Made, A. Sulistya, G. Made, y A. Sasmita, “Network Security Monitoring System on Snort with Bot Telegram as a Notification”, 2020. [En línea]. Disponible en: www.ijcat.com

[15] P. Schaub, “Managing and Monitoring”.

[16] K. Mounika, P. Venkateswara Rao, y A. Anbalagan, “Modified CNN Model for Network Intrusion Detection and Classification System Using Local Outlier Factor-based Recursive Feature Elimination”, *International Journal of Computer Network and Information Security*, vol. 17, no 1, pp. 82–91, feb. 2025, doi: 10.5815/ijcnis.2025.01.07.

[17] M. U. Meeting, “MikroTik Traffic Flow Network Monitoring / PRTG”, 2019. [En línea]. Disponible en: https://Tahandos.com

[18] M. Alkasassbeh y S. Al-Haj Baddar, “Intrusion Detection Systems: A State-of-the-Art Taxonomy and Survey”, *Arab J Sci Eng*, vol. 48, no 8, pp. 10021–10064, ago. 2023, doi: 10.1007/S13369-022-07412-1/METRICS.

[19] Y. Li, Q. Sun, D. Qin, K. Cheng, y Z. Li, “Power Control of a Modular Three-Port Solid-State Transformer with Three-Phase Unbalance Regulation Capabilities”, *IEEE Access*, vol. 8, pp. 72859–72869, 2020, doi: 10.1109/ACCESS.2020.2987075.

[20] I. H. Sarker, A. S. M. Kayes, S. Badsha, H. Alqahtani, P. Watters, y A. Ng, “Cybersecurity data science: an overview from machine learning perspective”, *J Big Data*, vol. 7, no 1, pp. 1–29, dic. 2020, doi: 10.1186/S40537-020-00318-5/FIGURES/3.

[21] L. Cong, J. Yu, y X. Ge, “Privacy preserving subgraph isomorphism query for dynamic graph database”, *Journal of Network and Computer Applications*, vol. 211, p. 103562, feb. 2023, doi: 10.1016/J.JNCA.2022.103562.

[22] S. Sadhwani, A. Verma, R. Muthalagu, y P. M. Pawar, “Network Intrusion Detection: A Study on Various Learning Approaches”, *Proceedings of 3rd IEEE International Conference on Computational Intelligence and Knowledge Economy, ICCIKE 2023*, pp. 161–166, 2023, doi: 10.1109/ICCIKE58312.2023.10131701.

[23] B. Nanu y N. Duhan, “Intrusion Detection in the Era of Machine Learning: A Critical Survey of Algorithms and Evaluation Practices”, *Int J Comput Appl*, vol. 187, no 21, pp. 50–57, jul. 2025, doi: 10.5120/IJCA2025925297.

[24] “A Guide to HIDS- Host-Based Intrusion Detection Systems - Sapphire.net”. Accedido: 18 de agosto de 2025. [En línea]. Disponible en: https://www.sapphire.net/blogs-press-releases/hids/

[25] “Understanding HIDS: Advantages and Limitations in Cybersecurity.” Accedido: 17 de agosto de 2025. [En línea]. Disponible en: https://guardiandigital.com/resources/blog/host-based-intrusion-detection-systems-cyber-threat-protection

[26] T. Xu, X. Ge, y C. Shao, “PMkR: Privacy-preserving multi-keyword top-k reachability query”, *Comput Secur*, vol. 156, sep. 2025, doi: 10.1016/j.cose.2025.104525.

[27] S. Elena, “UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA FACULTAD DE SISTEMAS Y TELECOMUNICACIONES”.

[28] H. Hindy *et al.*, “A Taxonomy of Network Threats and the Effect of Current Datasets on Intrusion Detection Systems”, *IEEE Access*, vol. 8, pp. 104650–104675, 2020, doi: 10.1109/ACCESS.2020.3000179.

[29] H. Liazid, M. Lehsaini, y A. Liazid, “Data transmission reduction using prediction and aggregation techniques in IoT-based wireless sensor networks”, *Journal of Network and Computer Applications*, vol. 211, feb. 2023, doi: 10.1016/j.jnca.2022.103556.

[30] W. A. H. Salman y C. H. Yong, “Overview of the CICIoT2023 Dataset for Internet of Things Intrusion Detection Systems”, *Mesopotamian Journal of Big Data*, vol. 2025, pp. 50–60, ene. 2025, doi: 10.58496/MJBD/2025/004.

[31] “Understanding Confusion Matrix | by Sarang Narkhede | Towards Data Science”. Accedido: 26 de noviembre de 2024. [En línea]. Disponible en: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

[32] M. Ali *et al.*, “Effective network intrusion detection using stacking-based ensemble approach”, *Int J Inf Secur*, vol. 22, no 6, pp. 1781–1798, dic. 2023, doi: 10.1007/s10207-023-00718-7.

[33] “MikroTik Routers and Wireless - Products: CRS310-1G-5S-4S+IN”. Accedido: 18 de agosto de 2025. [En línea]. Disponible en: https://mikrotik.com/product/crs310_1g_5s_4s_in

[34] “Página no encontrada - Documentación de MikroTik”. Accedido: 11 de diciembre de 2025. [En línea]. Disponible en: https://help.mikrotik.com/docs/download/pdf/manual_7_0.pdf

[35] “¿Qué es Docker Compose? Instalación y comandos”. Accedido: 27 de noviembre de 2025. [En línea]. Disponible en: https://intellipaat.com/blog/docker-compose/

[36] “Grafana→ Loki → Promtail: Proyecto completo de principio a fin | por Ghazanfar Ali | Medium”. Accedido: 29 de noviembre de 2025. [En línea]. Disponible en: https://ghazanfaralidevops.medium.com/grafana-loki-promtail-complete-end-to-end-project-d698aaa636d6