Polo del Conocimiento

Pol. Con. (Edición núm. 99) Vol. 9, No 10
Octubre 2024, pp. 647-670
ISSN: 2550 - 682X
DOI: https://doi.org/10.23857/pc.v9i10.8136

Uso de Software Libre y Machine Learning para mejorar la Detección de Intrusos en una Red

Using Free Software and Machine Learning to Improve Intrusion Detection in a Network

Utilização de Software Livre e Aprendizagem Automática para melhorar a Detecção de Intrusões numa Rede

Luis Eduardo Carrizo-Garcia [I]
luis.carrizogarcia7414@upse.edu.ec
https://orcid.org/0000-0002-9656-9917

María Daniela Álvarez-Galarza [II]
malvarez@upse.edu.ec
https://orcid.org/0009-0004-2209-0606

Correspondencia: luis.carrizogarcia7414@upse.edu.ec

Ciencias Técnicas y Aplicadas
Artículo de Investigación

*** Recibido:** 12 de agosto de 2024 ***Aceptado:** 24 de septiembre de 2024 *** Publicado:** 07 de octubre de 2024

I. Universidad Estatal Península de Santa Elena, Ecuador.
II. Universidad Estatal Península de Santa Elena, Ecuador.

Resumen

Este estudio integra técnicas de Machine Learning (ML) con el sistema de detección de intrusiones Snort para mejorar la identificación de ataques DDoS. El objetivo es reducir los falsos positivos y aumentar la precisión en la detección de amenazas en redes complejas. El método consistió en entrenar un modelo Random Forest utilizando el dataset CICIDS2017 y luego implementarlo junto a Snort en un entorno de red controlado. Los resultados mostraron un aumento en la precisión del 52.8% al 70.71%, y en la exactitud del 50.8% al 65.68%, con un incremento del F1-Score de 64.5% a 78.42%. Estos hallazgos demuestran que la integración de ML con Snort mejora significativamente la capacidad de detección y mitigación de incidentes en tiempo real. Se recomienda investigar el uso de otros algoritmos de ML y probar en diferentes escenarios para continuar optimizando el sistema.

Palabras clave: machine learning; detección de intrusiones; Snort, DDoS, ciberseguridad.

Abstract

This study integrates Machine Learning (ML) techniques with the Snort intrusion detection system to improve the identification of DDoS attacks. The goal is to reduce false positives and increase the accuracy of threat detection in complex networks. The method consisted of training a Random Forest model using the CICIDS2017 dataset and then deploying it alongside Snort in a controlled network environment. The results showed an increase in precision from 52.8% to 70.71%, and in accuracy from 50.8% to 65.68%, with an increase in the F1-Score from 64.5% to 78.42%. These findings demonstrate that integrating ML with Snort significantly improves the ability to detect and mitigate incidents in real time. It is recommended to investigate the use of other ML algorithms and test in different scenarios to continue optimizing the system.

Keywords: machine learning; intrusion detection; Snort, DDoS, cybersecurity.

Resumo

Este estudo integra técnicas de Machine Learning (ML) com o sistema de deteção de intrusão Snort para melhorar a identificação de ataques DDoS. O objetivo é reduzir os falsos positivos e aumentar a precisão da deteção de ameaças em redes complexas. O método consistiu em treinar um modelo Random Forest utilizando o conjunto de dados CICIDS2017 e depois implementá-lo juntamente

com o Snort num ambiente de rede controlado. Os resultados mostraram um aumento da precisão de 52,8% para 70,71%, e da exatidão de 50,8% para 65,68%, com um aumento do F1-Score de 64,5% para 78,42%. Estas descobertas demonstram que a integração do ML com o Snort melhora significativamente a deteção de incidentes em tempo real e as capacidades de mitigação. Recomenda-se investigar a utilização de outros algoritmos de ML e testar em diferentes cenários para continuar a otimizar o sistema.

Palavras-chave: aprendizagem automática; deteção de intrusão; Snort, DDoS, cibersegurança.

Introducción

En el entorno actual de la ciberseguridad, los sistemas de información y las telecomunicaciones se enfrentan a amenazas y ataques cibernéticos cada vez más sofisticados. Por lo que se requiere contar con una Arquitectura de Ciberseguridad robusta, dentro de esta se encuentran los denominados sistemas de detección de intrusiones (IDS), los cuales desempeñan un papel importante en la protección contra estas amenazas y ataques cibernéticos (Leiva, 2015). Sin embargo, una limitación común de muchos IDS es su dependencia de reglas predefinidas para detectar comportamientos anómalos y potencialmente maliciosos en el tráfico de la red. Esta autonomía los deja vulnerables a nuevos ataques que no saben que coinciden con las reglas almacenadas en sus bases de datos, lo que podría generar altos niveles de falsas alarmas o falsos positivos e incapacidad para detectar nuevas amenazas.

Según (IBM, 2024), un sistema de detección de intrusiones (IDS) actúa como un vigilante de la red, examinando constantemente el flujo de datos y los equipos conectados para identificar cualquier actividad que pueda indicar un ataque cibernético, un comportamiento anómalo o una violación de las normas de seguridad establecidas.

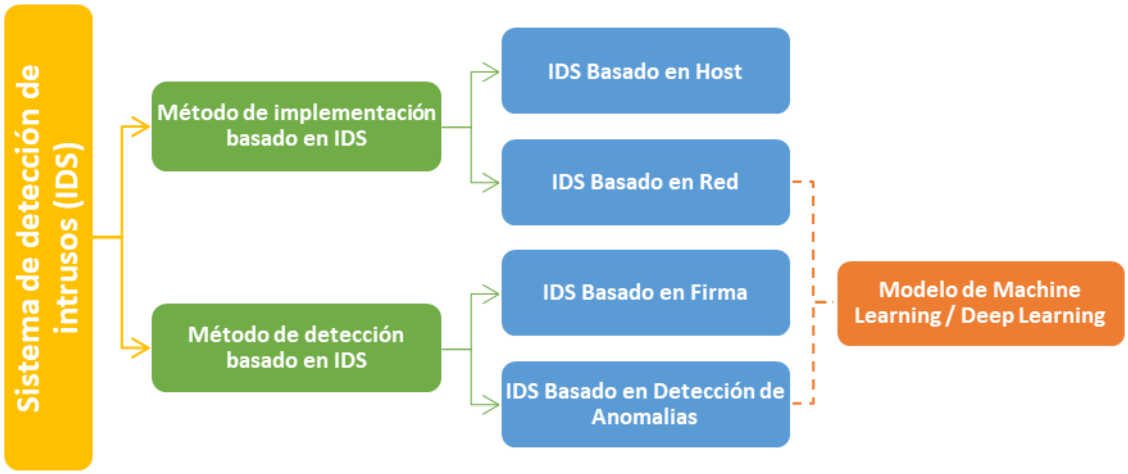
Desarrollo

Clasificación de los IDS

Según Ahmad et al., (2021), Los IDS se clasifican en dos categorías principales: por su implementación y por su método de detección. A su vez, cada una de estas categorías se subdivide en dos grupos adicionales: en la clasificación por implementación, encontramos los IDS basados en red y los basados en host; mientras que, en la clasificación por detección, se distinguen los IDS

basados en firmas y los basados en anomalías, a continuación, en la Figura 1 se puede visualizar dicha clasificación.

Figura 1: *Clasificación de los IDS.*



Fuente: *(Ahmad et al., 2021).*

Método de implementación basado en IDS

En cuanto el método de implementación los IDS se subclasifican como:

IDS Basado en Host (HIDS)

Están diseñados para operar de manera autónoma en cada dispositivo. Su función principal es monitorear continuamente las acciones que se llevan a cabo dentro del sistema, comparándolas con las normas de seguridad establecidas. Sin embargo, la necesidad de desplegar un HIDS en cada equipo que se desee proteger implica una carga computacional adicional en cada nodo, lo que puede comprometer el desempeño general del sistema de detección de intrusiones (Kabiri & Ghorbani, 2005).

IDS Basado en Red (NIDS)

Actúan como centinelas digitales, desplegados en puntos estratégicos de la red para salvaguardar todos los dispositivos conectados. Estos sistemas analizan de forma continua el flujo de datos en busca de patrones que indiquen un posible ataque o una violación de las políticas de seguridad (Ahmad et al., 2021).

Método de detección basado en IDS

En cuanto al método de detección se subdividen en:

Detección basada en firmas (SIDS)

Funciona mediante la creación de "huellas digitales" únicas para cada tipo de ataque. Estas huellas se almacenan en una base de datos y se comparan con el tráfico de la red en tiempo real. Si se encuentra una coincidencia, se genera una alerta (Axelsson, 2000).

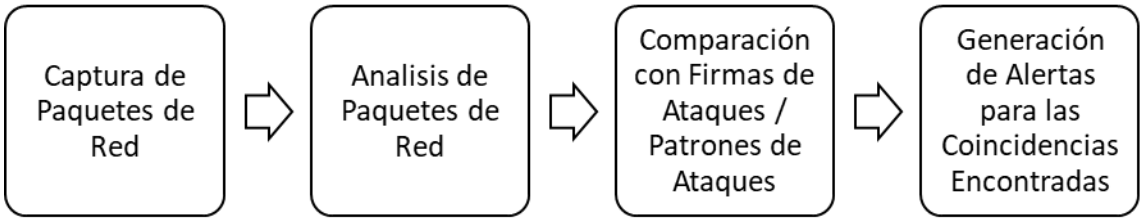
Detección basada en anomalías (AIDS)

Establece un patrón de comportamiento típico y saludable para la red. Cualquier acción que se desvíe significativamente de este patrón se considera sospechosa y potencialmente maliciosa (Filippo, 2000).

Arquitectura de un IDS

En la Figura 2, se observa la arquitectura básica de un Sistema de Detección de Intrusiones (IDS), destacando el proceso desde la captura y análisis de paquetes de red, hasta la comparación de estos con firmas o patrones de ataques conocidos, y la generación de alertas cuando se detectan coincidencias, lo cual permite identificar y responder a posibles amenazas en la red.

Figura 2: *Arquitectura de IDS.*



Fuente*: (Thakkar y Lohiya, 2020).*

Tipos de NIDS en el mercado

Esta investigación se enfoca en los NIDS porque representan una línea de defensa esencial en la protección de redes, interceptando tráfico malicioso antes de que pueda causar daños. Los NIDS

son especialmente útiles en entornos donde la protección de datos es primordial y donde la prevención de intrusiones es clave para la continuidad operativa.

Por un lado, Snort es el sistema de prevención de intrusiones de código abierto líder, que define la actividad maliciosa en la red mediante un conjunto de códigos para identificar y alertar sobre paquetes coincidentes, y puede interceptar estos paquetes en línea. Sus tres principales aplicaciones son rastreadoras de paquetes, registrador de paquetes para depuración y sistema completo de prevención de intrusiones (Snort, 2024). Zeek, anteriormente conocido como Bro, fue desarrollado por Vern Paxson en los años 1990 y renombrado en 2018 para reflejar su crecimiento. A diferencia de dispositivos de seguridad dinámicos, Zeek utiliza "sensores" para monitorear discretamente patrones de comunicación, creando registros precisos para análisis manual o herramientas SIEM (zeek, 2024). Por otro lado, el sistema de prevención denominado Suricata es un software de análisis de red y detección de amenazas de alto rendimiento, de código abierto, utilizado ampliamente en organizaciones para proteger sus activos (Suricata, 2024).

La importancia de este estudio radica en la necesidad urgente de mejorar los sistemas de detección de intrusos para hacer frente a la evolución constante de los ataques cibernéticos. Los ataques son cada vez más sofisticados y variados, y los métodos tradicionales basados en firmas no son suficientes para garantizar una protección adecuada. Si se integran técnicas de Machine Learning con el sistema de detección de intrusiones Snort, entonces se mejorará la capacidad de detección y mitigación de incidentes, reduciendo la tasa de falsos positivos.

Estado del arte

Una revisión de la literatura sobre la aplicación del aprendizaje automático (ML) en sistemas de detección de intrusiones (IDS) revela un creciente interés en su capacidad que mejora la precisión y escalabilidad de los IDS. Estudios previos demuestran que los algoritmos como máquinas de vectores de soporte (SVM), redes neuronales artificiales (ANN) y aprendizaje profundo (DL) son efectivos para detectar comportamientos anormales en el tráfico de red (Elshafie et al., 2020).

AbdulRaheem et al. (2024) encuentran mejoras notables en la precisión de detección de ataques DDoS al integrar ML con Snort y Zeek. El Aeraj y Leghris (2024) confirman que ML logra reducir las falsas alarmas y mejorar la detección de amenazas emergentes con Snort.

Otros estudios, como el de Fang y Liu (2011), muestran cómo los sistemas basados en ML pueden aprender continuamente, mejorando la detección en entornos dinámicos. Enigo et al. (2020)

proponen un marco híbrido de ML que mejora la detección y reducción de falsos positivos. Faizi et al. (2022) comparan técnicas de ML en Snort, encontrando que algunas ofrecen mejor precisión y velocidad en la detección de amenazas en tiempo real.

Amador et al. (2006) demuestran la capacidad de ML para detectar patrones anómalos en escaneos de puertos, mientras que Janampa Patilla et al. (2021) muestran mejoras en la seguridad de redes empresariales con ML y Snort. Guijarro Rodríguez et al. (2024) y Perdigón Llanes (2022) destan la efectividad de ML en la protección de infraestructuras críticas y redes empresariales, respectivamente.

Montes Vallejo (2023) y Chen y Lai (2023) revisan el uso de IA y ML en ciberseguridad, y la efectividad de ML en la detección de ataques DDoS en plataformas específicas. Coscia et al. (2024) mencionan que, mejoran la efectividad de IDS con reglas basadas en árboles de decisión, y Garba et al. (2024) proponen un marco que combina técnicas tradicionales de ML con Snort para proteger dispositivos IoT y controladores SDN (Software-Defined Networking) que gestionan el tráfico de red de manera programable y centralizada contra ataques DDoS.

Pantoja et al., (2021) comparan técnicas de ML, destacando la reducción de falsas alarmas y la mejora en la detección de amenazas. Estos estudios demuestran la capacidad de ML para mejorar los IDS, aunque la mayoría se centra en aspectos teóricos o escenarios de prueba, indicando una necesidad de estudios prácticos para validar la integración de ML y Snort en entornos reales.

SNORT

Para entender la posición destacada de ciertas herramientas en la detección y prevención de intrusiones, resulta relevante examinar estudios específicos que evalúan su desempeño bajo condiciones de ataque. En este sentido, el estudio realizado por Prabowo et al. (2023), confirma que Snort es la mejor opción entre las herramientas de Sistemas de Detección de Intrusiones (IDS) evaluadas debido a su excelente desempeño en la detección de intrusiones a nivel de red y sus capacidades de mitigación significativa de ataques de seguridad (DDoS).

Snort demuestra ser eficaz en el monitoreo de la calidad del servicio (QoS), mostrando un rendimiento excepcional en términos de rendimiento, baja latencia, baja fluctuación y bajas tasas de pérdida de paquetes durante los ataques Flood. Esta capacidad destaca la capacidad de Snort para preservar la integridad de la red y reducir el rendimiento en condiciones de alta carga,

posicionándola como la herramienta más eficaz para la protección DDoS según el análisis del estudio (Prabowo et al., 2023).

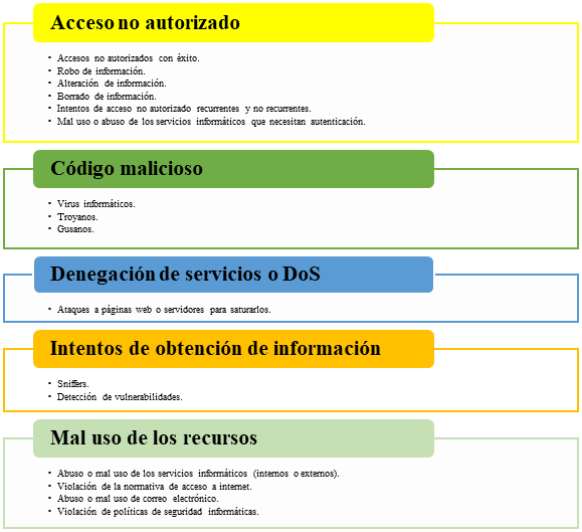
Snort procesa registros (logs) y eventos mediante la captura y registro del tráfico de red, generando alertas basadas en reglas configuradas. Los registros de eventos se pueden almacenar en una variedad de formatos, como archivos de texto, archivos binarios (pcap) o bases de datos (CSV), lo que facilita un análisis integral de los datos capturados (Snort, 2024). Además, Snort ofrece formas sencillas de administrar estos registros, permitiéndole almacenarlos localmente, cargarlos en un servidor remoto o integrarlos con un sistema de monitoreo centralizado de administración de eventos y seguridad (SIEM).

En comparación con otros IDS, Snort destaca por su variedad de reglas y firmas que son actualizadas periódicamente, lo que mejora enormemente la detección de amenazas conocidas. Su capacidad de personalización y flexibilidad le permite ajustar y crear reglas específicas adaptadas a las necesidades de la red administrada, algo que no siempre es posible con otros IDS. Además, la extensa comunidad de usuarios y desarrolladores de Snort aporta constantemente nuevo código y mejoras, proporcionando un sólido soporte gratuito. También permite integrarla con otras herramientas de seguridad lo que permite aumentar su capacidad de respuesta y eficiencia durante incidentes (Snort, 2024).

Incidentes de seguridad

Un incidente de seguridad es cualquier evento que compromete la integridad, confidencialidad o disponibilidad de la información (Chicano Tejada, 2023). Los incidentes de seguridad se pueden clasificar como se observa en la siguiente Figura 3

Figura 3: *Tipo e incidentes de seguridad.*

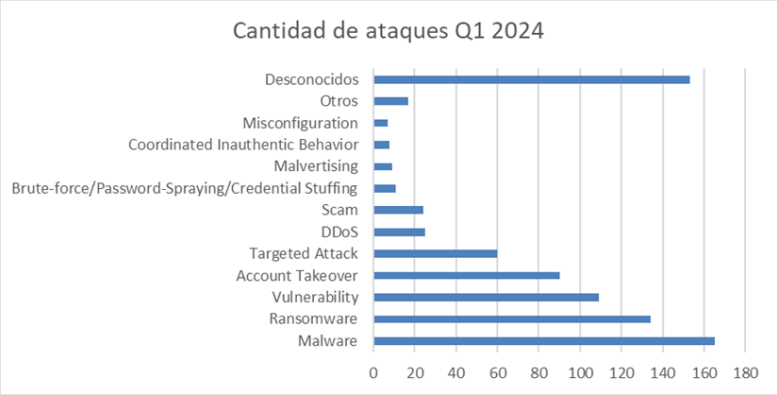


Fuente: *(Chicano Tejada, 2023).*

Estadísticas de los tipos de incidentes

La elección de hackmageddon.com como referencia para esta investigación se fundamenta en la confiabilidad y la continuidad de este proyecto, debido a que se ha documentado ataques cibernéticos desde hace más de una década. Además, la accesibilidad de los datos en formato bruto, ofrecidas directamente por el autor, permite que los investigadores obtengan información valiosa y actualizada sobre los tipos y frecuencias de ataques. Esta referencia es especialmente relevante debido a la falta de otras fuentes que mantengan una colección tan extensa y bien organizada de datos históricos de ataques cibernéticos.

Figura 4: *Cantidad de ataques Q1 2024.*



Fuente: *https://www.hackmageddon.com/2024/08/27/16-31-may-2024-cyber-attacks-timeline/*

Según la Figura 4 elaborado con los datos por hackmageddon.com sobre las técnicas de ataque más comunes en el primer trimestre de 2024, se observa que los Malware representan la mayor proporción de los ataques, constituyendo el 20% del total. Le sigue el Ransomware con un 17%, lo que refleja la prevalencia de estos ataques en el panorama de ciberseguridad actual.

La explotación de vulnerabilidades ocupa el tercer lugar con un 14%, seguida de los ataques de toma de control de cuentas con un 11% y los ataques dirigidos con un 8%. Por otro lado, los ataques DDoS y las estafas (Scam) representan 3% y 3% respectivamente, mostrando una menor incidencia en comparación con las técnicas dominantes (Passeri, 2024).

Dataset

CICIDS2017 (Canadian Institute for Cybersecurity Intrusion Detection System 2017) es un conjunto de datos ampliamente utilizado en la investigación y desarrollo de sistemas de detección de intrusiones (IDS). Fue creado por el Canadian Institute for Cybersecurity y diseñado para proporcionar una referencia estándar en el análisis y evaluación de técnicas de detección de intrusiones, incluyendo enfoques basados en machine learning y deep learning (Montes et al., 2023).

El dataset incluye varios tipos de ataques; sin embargo, este artículo se enfoca únicamente en los ataques DoS/DDoS. El conjunto de datos se reduce a un total de 79 atributos (incluyendo el atributo de clase) y 225745 instancias sin entrar en la etapa de procesamiento.

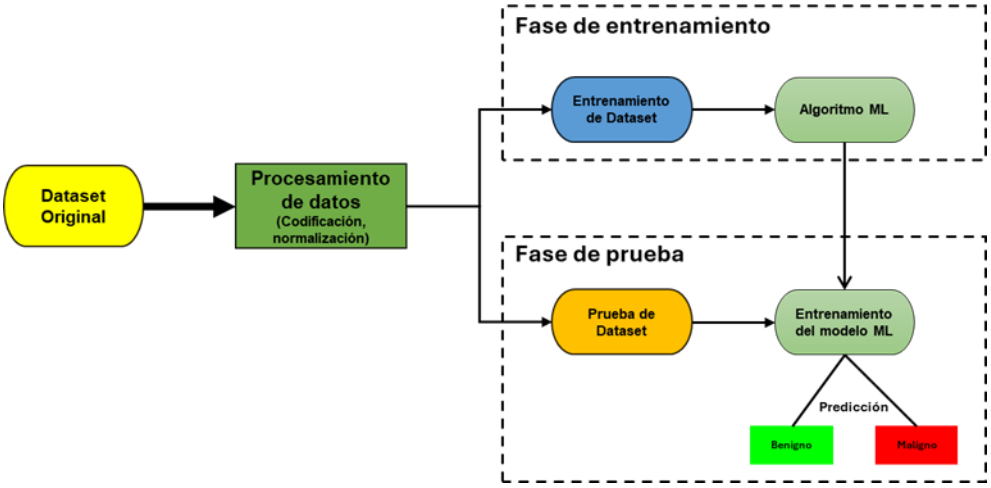
Un NIDS desarrollado con métodos de Machine Learning (ML) generalmente sigue tres fases principales:

- Fase 1: Procesamiento de datos.
- Fase 2: Entrenamiento.
- Fase 3: Prueba.

En la fase de preprocesamiento, el conjunto de datos se transforma al formato adecuado mediante codificación, normalización, y limpieza de datos faltantes o duplicados. Posteriormente, los datos se dividen en un conjunto de entrenamiento (80%) y un conjunto de prueba (20%). Durante la fase de entrenamiento, el algoritmo ML varía el tiempo de entrenamiento según el tamaño del conjunto de datos y la complejidad del modelo. Finalmente, el modelo entrenado se evalúa usando el

conjunto de prueba para predecir si el tráfico de red pertenece a la clase benigno (normal) o maligno (ataque) (Ahmad et al., 2021).

Figura 5: *Metodología de sistema de detección de intrusos basados en red (NIDS) y en aprendizaje automático (ML).*



Fuente: *(Ahmad et al., 2021).*

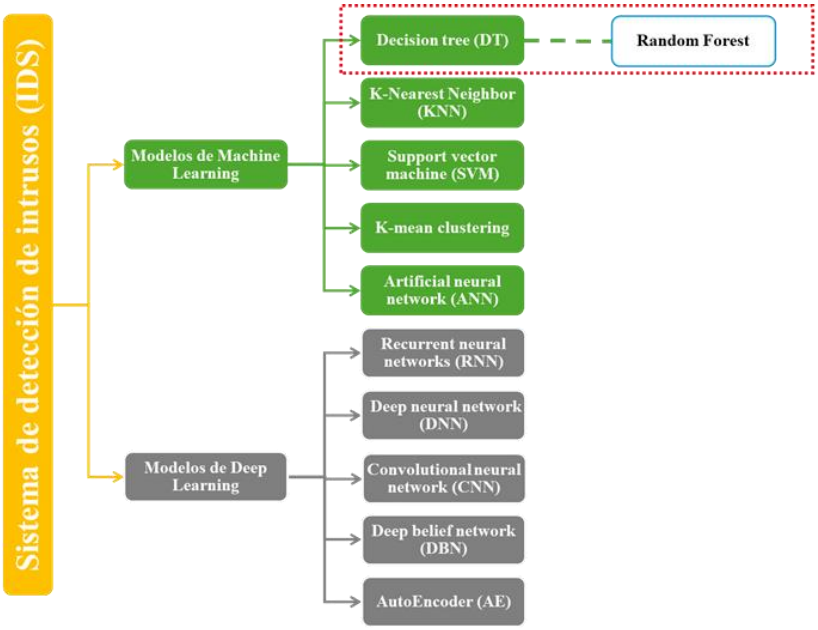
Métodos de IA para NIDS

En la Figura 6, se visualiza los algoritmos de aprendizaje automático (ML) y aprendizaje profundo (DL) utilizados en los sistemas de detección de intrusiones basados en red (NIDS). Los algoritmos de ML comúnmente utilizados incluyen árboles de decisión (DT), algoritmos de aprendizaje automático (K-vecino o KNN), redes neuronales artificiales (ANN), máquinas de vectores de soporte (SVM) y agrupación en clústeres K-Mean. En particular, el árbol de decisión clasifica y devuelve datos a través de una estructura de árbol que toma decisiones basadas en atributos, lo que facilita la interpretación del proceso de clasificación (Y. Xin et al., 2018). En este sentido, El algoritmo Random Forest basado en diferentes árboles de decisión mejora la precisión al combinar predicciones de diferentes modelos, lo que ayuda a reducir el riesgo de sobreajuste y mejorar la robustez (K. Rai et al., 2015).

Por otro lado, los algoritmos de Deep Learning (DL), como, redes neuronales recurrentes (RNN), redes neuronales para aprender representaciones de datos como autoEncoders (AE), redes neuronales profundas (DNN), redes neuronales convolucionales (CNN) y redes de creencias

profundas (DBN) (Ahmad et al., 2021). Son utilizados por los IDS para mejorar la capacidad de identificar y responder amenazas y/o ataques cibernéticos, ágil y eficazmente.

Figura 6: *Taxonomía para modelos de ML y DL basados en NIDS.*



Fuente: *(Ahmad et al., 2021).*

Evaluación de NIDS con la matriz de confusión

La matriz de confusión es una herramienta importante para evaluar el rendimiento de las técnicas de Machine Learning (ML) y Deep Learning (DL) utilizadas en los Sistemas de Detección de Intrusos (NIDS). Esta matriz bidimensional proporciona información sobre las predicciones del modelo en relación con los resultados reales, dividiendo los resultados en cuatro categorías (Ahmad et al., 2021):

1. **True Positive (TP):** Instancias correctamente clasificadas como ataques por el modelo.
2. **False Negative (FN):** Instancias de ataque que fueron clasificadas incorrectamente como normales.
3. **False Positive (FP):** Instancias normales clasificadas erróneamente como ataques.
4. **True Negative (TN):** Instancias normales correctamente identificadas como tales.

Precisión:

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (1)$$

Pol. Con. (Edición núm. 99) Vol. 9, No 10, Octubre 2024, pp. 647-670, ISSN: 2550 - 682X

Recall o Tasa de detección:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (2)$$

Tasa de falsas alarmas:

$$\text{Tasa de falsas alarmas} = \frac{FP}{FP+TN} \quad (3)$$

Tasa de negativos verdaderos:

$$\text{Tasa de negativos verdaderos} = \frac{TN}{TN+FP} \quad (4)$$

Accuracy:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

La diagonal de la matriz de confusión consta de valores True Positive (TP) y True Negative (TN), que representan la predicción correcta del modelo, es decir, cuando el modelo identificó correctamente un ataque o distribuyó el tráfico normal en el lugar correcto. Por otro lado, los valores fuera de la diagonal son False Negative (FN) y False Positive (FP), indican un error en el modelo cuando no detecta ataques (FN) o cuando no distribuye el tráfico normal (Deng et al., 2016).

Tabla 1: *Matriz de confusión.*

Predicción/Clase Real	**Predicción: Ataque**	**Predicción: Normal**
Real: Ataque	True Positive (TP)	False Negative (FN)
Real: Normal	False Positive (FP)	True Negative (TN)

Fuente: *(Deng et al., 2016).*

Este estudio tiene como objetivo integrar técnicas de aprendizaje automático (ML) con Snort para mejorar su capacidad de detección de intrusiones. Se busca desarrollar un sistema híbrido que combine las reglas predefinidas de Snort con la adaptabilidad y precisión de los modelos de ML. La investigación pretende demostrar que esta integración puede reducir significativamente las falsas alarmas o falsos positivos y mejorar la detección de nuevas amenazas, proporcionando una defensa más robusta y dinámica contra los ataques cibernéticos.

Metodología

En este estudio se utilizaron dos enfoques de investigación. Por un lado, el enfoque experimental se aplicó durante la fase de pruebas, en la cual los sistemas NIDS fueron evaluados en un entorno de red controlado mediante simulaciones de ataques reales y el uso de técnicas de aprendizaje automático, lo que permitió medir variables como la precisión y la sensibilidad. Por otro lado, el enfoque descriptivo permitió documentar el comportamiento de los NIDS a lo largo de los experimentos, realizando una comparación de las herramientas empleadas y su respuesta frente a diversas amenazas. Esto facilitó la evaluación de su eficacia y permitió sugerir mejoras para su implementación.

Preguntas de investigación

Q1: ¿Cómo afecta la integración de Machine Learning a la precisión de Snort en la detección de intrusos?

Q2: ¿Qué diferencias de comportamiento se observan en la detección de intrusiones con y sin la integración de Machine Learning?

Herramientas y software

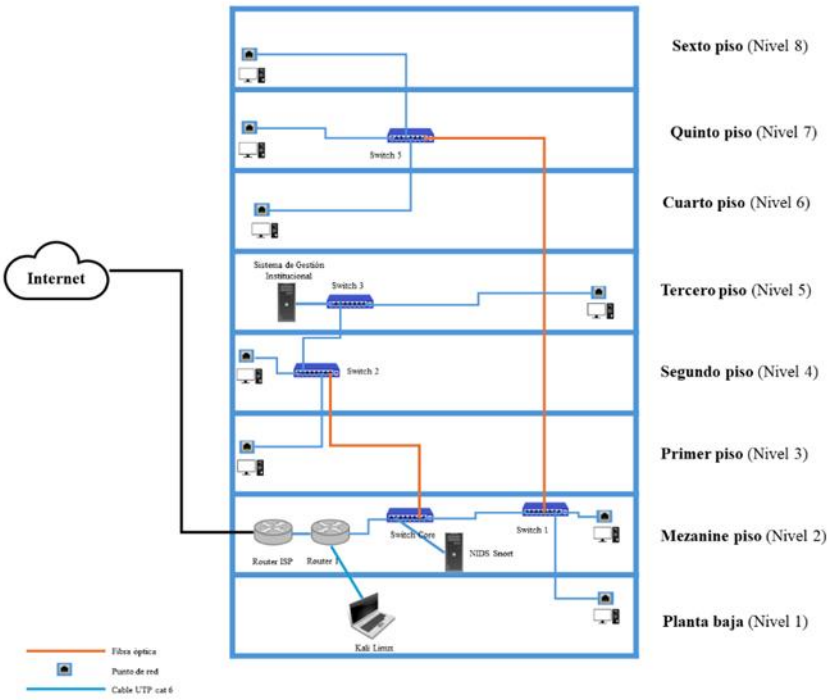
Se ha seleccionado a la herramienta Snort para este estudio, debido a que, es uno de los sistemas de detección de intrusiones más conocidos y confiables, y su capacidad para reconocer tráfico potencialmente malicioso a través de la red lo hace ideal para detectar amenazas (Adiwal et al., 2023).

Configuración de red

Para la configuración de red, se utiliza un servidor físico con un CPU de 2 núcleos y 16 GB de memoria RAM, ejecutando el sistema operativo Ubuntu Server 24.04.1, en el cual se encontraba instalado el NIDS Snort en su versión 2.9.15.1. La víctima fue una máquina física con el sistema operativo Ubuntu Server 24.04.1, mientras que el atacante operó desde un sistema con Kali Linux, ubicado en un segmento de red diferente. Todo el entorno se desarrolló en un escenario real pero controlado, con direcciones IP públicas expuestas a Internet, asignadas tanto a la máquina de la

víctima como al atacante, cada una en segmentos de red distintos. La configuración se detalla en la Figura 7.

Figura 7: *Topología física de la red.*



Fuente: *Elaboración propia.*

Tabla 2: *Configuración de cada sistema.*

Nombre del sistema	**CPU**	**Memoria**	**Sistema Operativo**
Sistema de Gestión Institucional (Victima)	Intel® Xeon® serie 5400	16 GB	Ubuntu Server 24.04.1
NIDS Snort 2.9.15.1	Intel® Xeon® serie 5100	16 GB	Ubuntu Server 24.04.1
Kali Linux	Intel® Core® I5	12 Gb	Kali Linux

Fuente: *Elaboración propia.*

Fases de procesamiento, entrenamiento y prueba del DataSet con Machine Learning

Los algoritmos de aprendizaje automático (ML), especialmente los árboles de decisión son importantes en los sistemas de detección de intrusiones en la red (NIDS). En este estudio se utilizó un modelo Random Forest con 100 árboles para detectar ataques DDoS, permitiendo la selección automática de características relevantes y evitando la sobrecarga mediante técnicas de poda, todo este proceso se realizó con la ayuda de la librería scikit-learn presente en el script en Python. Con ello, se procesó el grupo de datos del conjunto del DataSet CICIDS2017, y se separaron las variables predictivas y las variables objetivo normalizaron los valores utilizando StandardScaler. Esto resultó en un DataFrame listo para el entrenamiento del modelo.

Durante la fase de entrenamiento, los datos se dividieron en conjuntos de entrenamiento y prueba, utilizando el algoritmo Random Forest para entrenar el modelo y monitoreando su desempeño con una matriz de confusión e informes de clasificación. Este estudio incorporó parámetros claves como precisión, recall y F1-score, así como el tiempo de respuesta del modelo, asegurando así la medición de la eficiencia en tiempo real a través de un script en Python.

Pruebas con ataques DDoS

Pruebas sin ML en snort

En el proceso de prueba sin Machine Learning (ML), se utilizó Snort en su configuración tradicional basada en reglas estáticas para detectar posibles intrusiones. El tráfico de red fue generado simulando tráfico benigno normal en la red como ataques maliciosos DDoS con la ayuda de la herramienta Hping3 durante un tiempo definido. Snort analizó el tráfico en tiempo real utilizando las reglas predefinidas, que están diseñadas para detectar patrones específicos asociados a ataques conocidos. Las alertas generadas por Snort fueron registradas en el archivo de alerta snort.alert.fast. Está prueba se enfocó en observar cómo Snort maneja el tráfico sin la ayuda de técnicas de ML para identificar anomalías o intrusiones y, con un script en Python se evaluó el tráfico de red capturado para determinar su eficacia con la matriz de confusión.

Pruebas con ML en snort

La prueba de integración de Machine Learning (ML) en Snort consistió en tomar ya el modelo entrenado y probado con Random Forest y el DataSet de CICIDS2017 con el tráfico etiquetado en

formato CSV, los cuales fueron procesados para eliminar valores nulos y normalizar sus características. Luego, mediante un script en Python se analizó el tráfico de red en tiempo real, extrayendo las mismas características (columnas) utilizadas durante la fase de entrenamiento del modelo y clasificando los datos que registraron ataques DDoS. Se procedió con la puesta en marcha del script en Python y en paralelo se generó tráfico benigno y maligno como en la prueba anterior, el tráfico fue evaluado por el modelo ML, y cuando se detectó tráfico malicioso (DDoS), el sistema NIDS generó alertas y nuevas reglas dinámicas que se almacenaron en el archivo local.rules. Esto permitió automatizar la detección de ataque en tiempo real. Se realizó nuevamente la evaluación de la eficacia a través de la matriz de confusión con el script en Python.

Para la integración de Machine Learning en Snort, se utilizaron varias librerías de Python. La principal fue scikit-learn, que permitió entrenar y evaluar un modelo de Random Forest para la clasificación del tráfico de red. También se usó pandas para manipular los datos y procesar el conjunto de datos CICIDS2017, que se utilizó para entrenar el modelo. Finalmente, joblib facilitó la tarea de guardar y cargar el modelo entrenado para que pudiera reutilizarse en Snort sin necesidad de reentrenarlo cada vez. Estas herramientas fueron claves para mejorar la capacidad de detección de intrusiones de Snort.

Discusión de resultados

Desempeño del modelo en su fase de prueba

A través de la matriz de confusión se pudo visualizar el desempeño del modelo en su fase de prueba en donde se obtuvieron los resultados que se observan en la siguiente tabla:

Tabla 3: *Matriz de confusión en la fase de prueba del modelo.*

Predicción/Clase Real	Predicción: Ataque	Predicción: Normal
Real: Ataque	25657 (TP)	6 (FN)
Real: Normal	0 (FP)	18954 (TN)

Fuente: *Elaboración propia*.

El modelo realizó 18,954 predicciones correctas al clasificar adecuadamente los casos que no eran ataques DDoS (True Negative). No cometió ningún error al predecir ataques cuando en realidad no los había (False Positive). Sin embargo, se equivocó 6 veces al clasificar como no ataque DDoS

cuando en realidad sí lo era (False Negative). Además, acertó en 25,657 casos al identificar correctamente los ataques DDoS (True Positive).

En el informe de clasificación se pudo visualizar cómo se desempeñó el modelo de Machine Learning en la fase de prueba. La precisión (1.00) indica que todas las predicciones positivas fueron correctas, mientras que el recall (1.00) significa que el modelo identificó correctamente todas las instancias positivas. El F1-score (1.00) refleja un equilibrio perfecto entre precisión y recall. En el support se pudo visualizar, cuántas instancias reales hay en cada clase. La exactitud (1.00) indica que el modelo clasificó correctamente todas las instancias. Las métricas de promedio macro y promedio ponderado (ambos en 1.00) sugieren que el modelo tiene un rendimiento excelente en general, sin importar el tamaño de las clases. En conjunto, estos resultados indican que el modelo clasificó todas las instancias de manera correcta.

Tabla 4: *Informe del desempeño del modelo (fase 3 de prueba).*

	precision	**recall**	**F1-score**	**support**
False	1.00	1.00	1.00	18954
True	1.00	1.00	1.00	25663
Accuracy			1.00	44617
macro avg	1.00	1.00	1.00	44617
weighted avg	1.00	1.00	1.00	44617

Fuente: *Elaboración propia.*

Desempeño de Snort Sin ML

Se identificaron correctamente 22 instancias de tráfico benigno, clasificándolas como tal (True Negative). Sin embargo, cometió 142 errores al clasificar incorrectamente tráfico benigno como ataques (False Positive). Además, no detectó 33 ataques, clasificándolos como benignos (False Negative). Por otro lado, logró identificar correctamente 159 ataques como maliciosos (True Negative). Como se visualiza en la Tabla 5.

Tabla 5: *Matriz de confusión sin ML.*

Predicción/Clase Real	**Predicción: Ataque**	**Predicción: Normal**
Real: Ataque	159 (TP)	33 (FN)

Real: Normal	142 (FP)	22 (TN)

Fuente: *Elaboración propia.*

Como se observa en la Tabla 6, el desempeño sin ML alcanzó una precisión de 52.82%, lo que indica que, de todas las instancias clasificadas como ataques, poco más de la mitad realmente lo eran. La exactitud fue del 50.84%, lo que significa que un poco más de la mitad de las predicciones fueron correctas en general. El recall, o sensibilidad, fue del 82.81%, ello implica que, snort fue capaz de detectar la mayoría de los ataques reales. Finalmente, el F1-Score, que equilibra la precisión y el recall, fue de 64.50%, reflejando un desempeño moderado en la clasificación de ataques cibernéticos.

Tabla 6: *Informe de desempeño sin ML.*

	Resultados	**%**
Precision	0.528	52.8
Accuracy	0.508	50.8
Recall	0.828	82.8
F1-score	0.645	64.5

Fuente: *Elaboración propia.*

Desempeño de Snort con ML

Snort con ML clasificó correctamente 9 instancias de tráfico benigno como benignas (True Negative). Sin embargo, cometió 70 errores al clasificar incorrectamente tráfico benigno como ataques (False Positive). Además, no detectó 23 ataques, clasificándolos incorrectamente como benignos (False Negative). Por otro lado, logró identificar correctamente 169 ataques como maliciosos (True Positive).

Tabla 7: *Matriz de confusión con ML.*

Predicción/Clase Real	**Predicción: Ataque**	**Predicción: Normal**
Real: Ataque	169 (TP)	23 (FN)
Real: Normal	90 (FP)	9 (TN)

Fuente: *Elaboración propia.*

Con la integración de Machine Learning, Snort alcanzó una precisión del 70.71%, lo que significa que el sistema genera relativamente pocos falsos positivos. En cuanto a la exactitud fue del 65.68%,

lo que indica que más de la mitad de las predicciones fueron correctas. El recall fue del 88.02%, lo que refleja una alta capacidad del sistema para detectar la mayoría de los ataques. Finalmente, el F1-Score se ubicó en 78.42%, mostrando un buen equilibrio entre precisión y recall, lo que indica un desempeño efectivo en la clasificación de intrusiones.

Tabla 8: *Informe de desempeño con ML.*

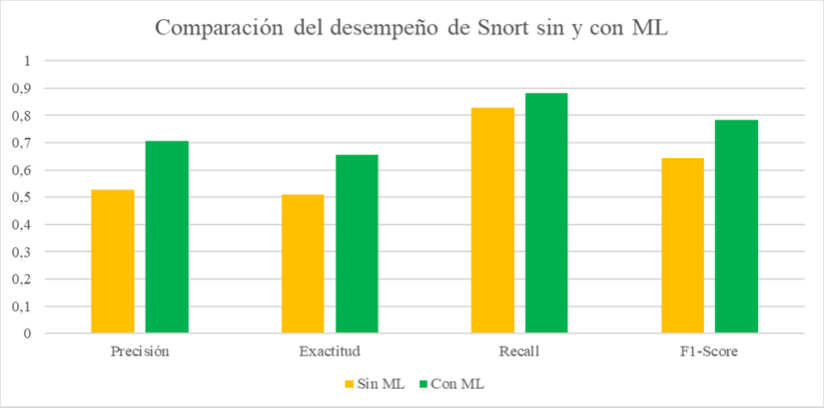
	Resultados	**%**
Precision	0.707	70.7
Accuracy	0.656	65.6
Recall	0.880	88.0
F1-score	0.784	78.4

Fuente: *Elaboración propia.*

Comparación del desempeño

Al comparar el sistema con y sin Machine Learning en Snort, se observan mejoras significativas. La precisión aumenta del 52.8% sin ML al 70.71% con ML, lo que indica que el sistema redujo considerablemente los falsos positivos. la exactitud se eleva del 50.8% al 65.68%, y demuestra que el sistema fue mucho más efectivo en la clasificación correcta de las instancias con ML. el recall se mantiene alto, experimenta una ligera mejora con la integración de ML. Finalmente, El F1-Score muestra un incremento notable, pasando de 64.5% a 78.42%, lo que refleja un mejor equilibrio entre la detección de ataques cibernéticos y la reducción de falsos positivos con ML.

Figura 8: *Comparación del desempeño de Snort sin y con ML*



Fuente: *Elaboración propia.*

Pol. Con. (Edición núm. 99) Vol. 9, No 10, Octubre 2024, pp. 647-670, ISSN: 2550 - 682X

Conclusiones

Los resultados obtenidos demuestran mejoras significativas en la detección de ataques DDoS al integrar Machine Learning en el sistema Snort. La precisión del modelo mejoró notablemente, pasando de un 52.8% a un 70.71%, lo que indica una reducción efectiva de los falsos positivos. El F1-Score también presentó un aumento significativo, reflejando un mejor equilibrio entre la detección de ataques y la clasificación de tráfico benigno. La exactitud general del sistema se incrementó del 50.8% al 65.68%, validando la eficacia de la integración de técnicas de aprendizaje automático.

Este estudio contribuye al campo de la ciberseguridad al mostrar cómo la implementación de Machine Learning en sistemas de detección de intrusiones como Snort puede mejorar significativamente la detección de amenazas y ataques cibernéticos, especialmente en entornos de red complejos donde es esencial minimizar tanto los falsos positivos como los falsos negativos. Al optimizar la precisión y el balance entre la detección de ataques cibernéticos y la reducción de errores, este trabajo aporta una solución práctica y escalable para mejorar la seguridad en redes.

Para investigaciones futuras, se sugiere explorar la combinación de diferentes algoritmos de Machine Learning para lograr un mayor nivel de precisión y recall. Además, sería valioso realizar pruebas en entornos más diversos, con diferentes tipos de ataques y volúmenes de tráfico, para evaluar el rendimiento del sistema en otras condiciones. También se podría considerar la implementación de técnicas de aprendizaje profundo para optimizar aún más la detección y respuesta a incidentes en tiempo real.

Referencias

1. Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1). https://doi.org/10.1002/ett.4150
2. AbdulRaheem, M., Oladipo, I. D., Imoize, A. L., Awotunde, J. B., Lee, C. C., Balogun, G. B., & Adeoti, J. O. (2024). Machine learning assisted snort and zeek in detecting DDoS attacks in software-defined networking. International Journal of Information Technology, 16(3), 1627-1643.

3. Adiwal, S., Rajendran, B., Shetty D., P., & Sudarsan, S. (2023). DNS Intrusion Detection (DID) — A SNORT-based solution to detect DNS Amplification and DNS Tunneling attacks. ELSEIVER, 1-10.
4. Amador, S., Arboleda, A. Y., & Bedón, C. (2006). Utilizando Inteligencia Artificial para la detección de Escaneos de Puertos.
5. Chen, C. L., & Lai, J. L. (2023). An Experimental Detection of Distributed Denial of Service Attack in CDX 3 Platform Based on Snort. Sensors, 23(13). https://doi.org/10.3390/s23136139
6. Chicano Tejada, E. (2023). Gestión de incidentes de seguridad informática. Málaga: IC.
7. Coscia, A., Dentamaro, V., Galantucci, S., Maci, A., & Pirlo, G. (2024). Automatic decision tree-based NIDPS ruleset generation for DoS/DDoS attacks. Journal of Information Security and Applications, 82. https://doi.org/10.1016/j.jisa.2024.103736
8. Deng, X., Liu, Q., Deng, Y., & Mahadevan, S. (2016). An improved method to construct basic probability assignment based on the confusion matrix for classification problem. sciencedirect, 250-261.
9. El Aeraj, O., & Leghris, C. (2024). Analysis of the SNORT intrusion detection system using machine learning. International Journal of Information Science and Technology, 8(1), 1-9.
10. Elshafie, H. M., Mahmoud, T. M., & Ali, A. A. (2020). An efficient Snort NIDSaaS based on danger theory and machine learning. Applied Mathematics, 14(5), 891-900.
11. Enigo, V. S. F., Ganesh, K. T., Raj, N. N. V., & Sandeep, D. (2020). Hybrid intrusion detection system for detecting new attacks using machine learning. In 2020 5th International Conference on Communication and Electronics Systems (ICCES) (pp. 567-572). IEEE.
12. Faizi, A. H. N., Priambodo, D. F., & Rahmawati, F. D. (2022). Comparison of machine learning techniques on Snort for predicting realtime DoS and probe attack. In 2022 International Conference on Informatics Multimedia Cyber and Information System (ICIMCIS) (pp. 224-229). IEEE.
13. Fang, X., & Liu, L. (2011). Integrating artificial intelligence into Snort IDS. In 2011 3rd International Workshop on Intelligent Systems and Applications (pp. 1-4). IEEE.

14. Filippo, N. (2000). Comparing local search with respect to genetic evolution to detect intrusions in computer networks. Proceedings of the 2000 Congress on Evolutionary Computation.
15. Garba, U. H., Toosi, A. N., Pasha, M. F., & Khan, S. (2024). SDN-based detection and mitigation of DDoS attacks on smart homes. Computer Communications, 221, 29–41. https://doi.org/10.1016/j.comcom.2024.04.001
16. Guijarro Rodríguez, A. A., Jácome Morales, G. C., Gonzalez Mestanza, V., Terán Zurita, E., & Torres Martínez, D. E. (2024). 2022. Detección de amenazas de seguridad en una red corporativa utilizando algoritmos de machine learning. Serie Científica de La Universidad de Las Ciencias Informáticas, 15, 1–11.
17. IBM. (15 de julio de 2024). https://www.ibm.com/. Obtenido de https://www.ibm.com/es-es/topics/intrusion-detection-system
18. Janampa Patilla, H., Huamani Santiago, H. L., & Meneses Conislla, Y. (2021). Snort Open Source como detección de intrusos para la seguridad de la infraestructura de red Snort Open Source as intrusion detection for network infrastructure security Hubner Janampa Patilla 1* https://orcid.org/0000-0003-3110-194X. Revista Cubana de Ciencias Informáticas, 15(3). http://rcci.uci.cuPág.55-73https://orcid.org/0000-0002-8197-9956YudithMenesesConislla3https://orcid.org/0000-0002-7646-5512
19. Leiva, E. A. (2015). Estrategias nacionales de ciberseguridad: Estudio comparativo basado en enfoque top-down desde una visión global a una visión local. Archivo de la Revista Latinoamericana de Ingeniería de Software, 3(4), 161-176.
20. Montes Vallejo, C. F. (2023). Inteligencia Artificial y el Aprendizaje Automático en la Ciberseguridad.
21. Montes Gil, J. A., Isaza Cadavid, G., & Duque Méndez, N. D. (2023). Efecto de la selección de atributos en el desempeño de un IDS basado en machine learning para detección de intrusos en ataques DDoS. South Florida Journal of Developmen, 1-11.
22. Pantoja, N. D., Donado, S. A., Villalba, K. M., & Martínez Flor, E. U. (2021). 2021. Determinación de técnica de inteligencia para la detección de un tipo de ataque en un IDS. Revista Ibérica de Sistemas e Tecnologias de Informação, 317–329.
23. Passeri, P. (09 de 07 de 2024). https://www.hackmageddon.com/. Obtenido de https://www.hackmageddon.com/2024/07/09/q1-2024-cyber-attacks-statistics/

24. Perdigón Llanes, R. (2022). Suricata como detector de intrusos para la seguridad en redes de datos empresariales. Ciencia UNEMI, 15(39), 44–53. https://doi.org/10.29076/issn.2528-7737vol15iss39.2022pp44-53p
25. Prabowo, W. A., Fauziah, K., Nahrowi, A. S., Faiz, M. N., & Muhammad, A. W. (2023). Strengthening Network Security: Evaluation of Intrusion Detection and Prevention Systems Tools in Networking Systems. International Journal of Advanced Computer Science and Applications, 1-10.
26. Rai, K., Devi, M. S., & Guleria, A. (2015). Decision tree based algorithm for intrusion detection. International Journal of Advanced Networking and Applications, 7(4), 2828.
27. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp, 1, 108-116.
28. Snort. (2024). https://www.snort.org/. Obtenido de https://www.snort.org/
29. Suricata. (2024). https://suricata.io/. Obtenido de https://suricata.io/
30. Y. Xin et al., "Machine Learning and Deep Learning Methods for Cybersecurity," in IEEE Access, vol. 6, pp. 35365-35381, 2018, doi: 10.1109/ACCESS.2018.2836950.
31. Zeek. (2024). https://zeek.org/. Obtenido de https://zeek.org/