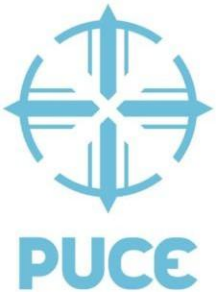


PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

ESCUELA DE HÁBITAT, INGENIO Y CREATIVIDAD

TRABAJO DE INTEGRACIÓN CURRICULAR PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN

SISTEMA WEB BASADO EN APRENDIZAJE AUTOMÁTICO PARA LA DETECCIÓN TEMPRANA DE ATAQUES DE DENEGACIÓN DE SERVICIO (DOS) EN REDES WIFI

DIEGO CARLOS PABON RAMIREZ

TUTOR: MGS. DARWIN PILLO

IBARRA – ECUADOR

JULIO, 2025

Ibarra, 15 de agosto de 2025

CERTIFICACIÓN TUTOR

En mi calidad de Tutor del Trabajo de Titulación titulado: Sistema web basado en aprendizaje automático para la detección temprana de ataques de denegación de servicio (DOS) en redes wifi, presentado por el estudiante Diego Carlos Pabón Ramírez con cédula de ciudadanía N°1003459235, para obtener el Título de Ingeniero en Tecnologías de la Información.

Certifico que el trabajo cumple con todos los parámetros establecidos, mediante el cual el estudiante demuestra el desarrollo de competencias en el campo de conocimiento de su profesión con un nivel de argumentación coherente, para ser sometido a la evaluación por parte de los lectores.

Adicionalmente, se adjunta el certificado de porcentaje de originalidad de TURNITIN.


12/8/25, 8:20 Turnitin - Informe de Originalidad - TESIS_PABON_9-08-2025_Final.docx

Turnitin Informe de Originalidad

Procesado el: 12-ago-2025 08:15 -05
Identificador: 2710275037
Número de palabras: 20755
Entregado: 2

TESIS_PABON_9-08-2025_Final.docx Por DIEGO CARLOS PABON RAMIREZ

Índice de similitud 7%

Similitud según fuente
Fuentes de Internet: 6%
Publicaciones: 1%
Trabajos del estudiante: 4%

Coincidencia del 1% (Internet desde 18-mar-2025)
https://repositorio.puce.edu.ec/server/api/core/bitstreams/1f1f74a5-2e4d-432b-be2a-913463826fce/content

Coincidencia del < 1% (trabajos de los estudiantes desde 11-jun-2024)
Submitted to Pontificia Universidad Catolica del Ecuador - PUCE on 2024-06-11

Coincidencia del < 1% (trabajos de los estudiantes desde 16-jul-2025)
Submitted to Pontificia Universidad Catolica del Ecuador - PUCE on 2025-07-16

Darwin Pillo G.

Firmado digitalmente por Darwin Pillo G.
DN: cn=Darwin Pillo G. gn=Darwin Pillo G. c=EC Ecuador l=EC Ecuador o= Escuela de Arquitectura, Ingeniería, Diseño y Artes ou=PUCE Ibarra e=dmpillo@pucesi.edu.ec
Motivo:Apruebo este documento
Ubicación:Ibarra - Ecuador
Fecha:2025-08-14 11:38-05:00

(f):
Mgs. Darwin Marcelo Pillo Guanoluisa
TUTOR DE TRABAJO
C.C.: 1003319660

PÁGINA DE APROBACIÓN DEL TRIBUNAL

El tribunal examinador, aprueba el presente trabajo en nombre de la Pontificia Universidad Católica del Ecuador Ibarra:

Darwin Pillo G.

Firmado digitalmente por Darwin Pillo G.
DN: cn=Darwin Pillo G. gn=Darwin Pillo G. c=EC Ecuador l=EC Ecuador o=Escuela de Arquitectura, Ingeniería, Diseño y Artes ou=PUCE Ibarra e=dmpillo@pucesi.edu.ec
Motivo:Apruebo este documento
Ubicación:Ibarra - Ecuador
Fecha: 2025-08-14 11:38:49-00'

(f): ……

Mgs. Darwin Marcelo Pillo Guanoluisa

C.C.: 1003319660

Laura Guerra Torrealba

Firmado digitalmente por Laura Guerra Torrealba
Fecha: 2025.08.14 12:33:14 -05'00'

(f):…… ……

PhD. Guerra Torrealba Laura Rosa

C.C.: 1757842784

Dulce Milagro Rivero Albarrán

Firmado digitalmente por Dulce Milagro Rivero Albarrán
Fecha: 2025.08.15 09:03:34 -05'00'

(f):……

PhD. Rivero Albarrán Dulce Milagro

C.C.: 1757608961

ACTA DE CESIÓN DE DERECHOS

Yo, *Diego Carlos Pabón Ramírez*, declaro conocer y aceptar la disposición del Art. 165 del Código Orgánico de Economía Social de los Conocimientos, Creatividad e Innovación, que manifiesta textualmente: “Se reconoce facultad de los autores y demás titulares de derechos de disponer de sus derechos o autorizar las utilizaciones de sus obras o prestaciones a título gratuito y oneroso, según las condiciones que determinen. Esta facultad podrá ejercerse mediante licencias libres, abiertas y otros modelos alternativos de licenciamiento o la renuncia”.

Ibarra, 15 de agosto de 2025

DIEGO CARLOS PABON RAMIREZ
Firmado digitalmente por DIEGO CARLOS PABON RAMIREZ
Fecha: 2025.08.14 11:35:17 -05'00'

(f): _____

DIEGO CARLOS PABÓN RAMÍREZ

C.C.: 1003459235

AUTORIA

Yo, *DIEGO CARLOS PABÓN RAMÍREZ*, portador@ de la cedula de ciudadanía N° 1003459235, declaro que el presente trabajo de investigación es de total responsabilidad de la autor@, y eximo expresamente a la Pontificia Universidad Católica del Ecuador Ibarra de posibles reclamos o acciones legales.

DIEGO CARLOS PABON RAMIREZ
Firmado digitalmente por DIEGO CARLOS PABON RAMIREZ
Fecha: 2025.08.14 11:35:43 -05'00'

(f):……..

DIEGO CARLOS PABÓN RAMÍREZ

C.C.: 1003459235

DEDICATORIA Y AGRADECIMIENTOS

A mi madre, Mery, cuyo amor incondicional, sacrificio y apoyo constante han sido el motor de mi vida y mi pilar fundamental. Gracias por cada enseñanza, cada abrazo y por creer siempre en mí. Eres mi mayor inspiración.

A mi segunda madre, Myriam, por tu cariño, tu guía y por el apoyo en mi crecimiento personal.

A mis primos hermanos, Daniel y Carolina, compañeros de vida y cómplices. Gracias por su amistad, su aliento y por compartir cada paso de esta aventura. Su apoyo ha sido invaluable.

A Andrea, por tu apoyo incondicional durante esta etapa tan significativa. Tu constancia y aliento fueron lo más importante para mí.

A mis queridos amigos Ángel, Marco y Giuliana, por su amistad, por el apoyo en cada desafío. Gracias por ser esa familia que elegí y por la motivación que me brindaron para no desistir.

A la Universidad Católica Del Ecuador Sede Ibarra, por brindarme toda los conocimientos y enseñanzas, también por permitir realizar mi trabajo para obtener este gran logro.

ÍNDICE DE CONTENIDOS

Contenido

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En la era digital, la conectividad inalámbrica es fundamental para el desarrollo académico, laboral y social, y las redes Wi-Fi facilitan el acceso a la información. Por lo tanto, estas redes con el tiempo se han convertido en objetivos de ataques cibernéticos, especialmente de Denegación de Servicio (DoS), lo que buscan es saturar los recursos de la red para interrumpir sus servicios. Los ataques representan un riesgo para las instituciones como la Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCE-I), donde la correcta estabilidad tecnológica es vital para el correcto funcionamiento de las actividades académicas y administrativas.

Con el objetivo de fortalecer la ciberseguridad de la institución, esta investigación desarrolló un sistema web para la detección temprana de ataques DoS en redes Wi-Fi, utilizando técnicas de aprendizaje automático. Para ello, se tomó como base un Dataset de tráfico de red, a partir del cual se entrenó al modelo, lo que permitió el desempeño del sistema en escenarios representativos.

La metodología combinó el proceso KDD (Knowledge Discovery in Databases) para el entrenamiento del modelo de aprendizaje automático y la metodología ágil XP (Extreme Programming) para el diseño del sistema. Los datos fueron transformados al formato Parquet para optimizar su carga y análisis, para lo cual se entrenó el modelo Random Forest que fue capaz de clasificar el tráfico como benigno o como otros ataques DoS.

El sistema está estructurado en módulos que incluyen la gestión de archivos de tráfico, análisis predictivo, visualización de métricas (curvas ROC, Precision-Recall), análisis estadístico (Histogramas y boxplots) y un registro detallado de resultados. La interfaz web permite que a los usuarios cargar los archivos, observar resultados gráficos y analizar características clave del tráfico, teniendo así la facilidad de interpretación y toma de decisiones para el personal administrativo.

El enfoque permitió detectar con precisión patrones anómalos en el tráfico y presentar los resultados de manera accesible. De esta forma se cuenta con una herramienta eficaz para prevenir interrupciones causadas por el ataque DoS, asegurando la continuidad académica y la protección de su infraestructura tecnológica.

Palabras clave: redes Wi-Fi, detección temprana, aprendizaje automático, KDD, XP, seguridad informática, tráfico de red, Random Forest, sistema.

ABSTRACT

In the digital era, wireless connectivity is fundamental for academic, labor, and social development, and Wi-FI networks facilitate access to information. Therefore, these networks have over time become targets of cyber-attacks, especially Denial of Service (DoS), which seek to saturate the network´s resources to interrupt its services. The attacks represent a risk for institutions such as the Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCE-I), where the correct technological stability is vital for the proper functioning of academic and administrative activities.

With the aim of strengthening the cybersecurity of the institution, this research developed a web system for the early detection of DoS attacks on Wi-Fi networks, using machine learning techniques. For this, a dataset of network traffic was used as a basis, for which the model was trained, allowing the system to perform in representative scenarios.

The methodology combined the KDD (Knowledge Discovery in Databases) process for training the machine learning model and the agile XP (Extreme Programming) methodology for system design. The data was transformed into Parquet format to optimize its loading and analysis, for which the Random Forest model was trained, capable of classifying traffic as benign or as other DoS attacks.

The system is structured in modules that include traffic file management, predictive analysis, visualization of metric (ROC curves, Precision Recall), statistical analysis (Histograms and Boxplot), and a detailed record of results. The web interface allows users to upload files, observe graphical results, and analyze key characteristics of the traffic, this facilitating interpretation and decision-making for administrative personnel. The approach allowed for the precise detection of anomalous patterns in the traffic and presented the results in an accessible manner. In this way, there is an effective tool to prevent interruptions caused by the DoS attack, ensuring academic and protection of this technological infrastructure.

Keywords: Wi-Fi networks, early detection, machine learning, KDD, XP, computer security, network traffic, Random Forest, system.

INTRODUCCIÓN

En la era digital actual, las redes inalámbricas Wi-Fi se han convertido en una de las infraestructuras críticas para la comunicación y la obtención de información, en entornos domésticos y empresariales. Sin embargo, estas redes también están expuestas a diversas amenazas cibernéticas, entre las que los ataques de Denegación de Servicio (DoS) son uno de los más comunes. Estos ataques lo que buscan es sobrecargar los recursos de la red, impidiendo su disponibilidad y afectando la calidad del servicio para los usuarios legítimos. Cuando estos ataques son realizados de manera distribuida, su impacto se multiplica, comprometiendo así la infraestructura de las redes, así como la confidencialidad e integridad de la información que esta transmite.

La detección temprana de los ataques DoS en redes Wi-Fi son cruciales para poder mitigar sus efectos y restaurar la normalidad de las operaciones lo antes posible. Debido a la evolución constante de las técnicas de ataque y la complejidad de los entornos de red, los sistemas tradicionales para la detección de intrusos a menudo resultan insuficientes. En este contexto, el aprendizaje automático se ha posicionado como una herramienta prometedora para la identificación de patrones anómalos en el tráfico de red y la clasificación de comportamientos maliciosos.

La Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCE-I), es una institución académica de prestigio, comprometida con la innovación tecnológica y la formación de profesionales altamente capacitados. La Unidad de Tecnologías de la Información de la PUCE-I juega un rol clave en la gestión y protección de la infraestructura tecnológica de la universidad. Sin embargo, la PUCE-I enfrenta desafíos relacionados con la seguridad de su red Wi-Fi, especialmente ante el riesgo creciente de ataque DoS que podrían interrumpir las actividades académicas y administrativas.

La investigación tuvo como propósito diseñar un sistema para innovar, monitorear, y detectar tempranamente los ataques DoS en redes Wi-Fi en un entorno de prueba dentro de la PUCE-I. Además de fortalecer la seguridad de la infraestructura tecnológica de dicha institución, mejorando la ciberseguridad y garantizando la disponibilidad de los servicios de la red.

Para alcanzar este propósito, se presentan los objetivos del estudio.

Objetivo General

- Desarrollar un sistema web de monitorización y detección temprana de ataques de denegación de servicio (DoS) en redes Wi-Fi, que utilice técnicas de aprendizaje automático para identificar patrones de tráfico malicioso y mitigar estas amenazas de manera anticipada, asegurando así la disponibilidad y seguridad de la infraestructura de red frente a posibles ataques.

Objetivos Específicos

- Realizar un análisis exhaustivo de la situación actual de los sistemas de detección de ataques DoS en redes Wi-Fi, identificando sus características y limitaciones que permita establecer un marco teórico sólido para el desarrollo de un sistema web basado en aprendizaje automático.
- Seleccionar el algoritmo de aprendizaje automático más adecuado para la detección temprana de ataques DoS en redes Wi-Fi, considerando su capacidad para identificar patrones de tráfico malicioso y su efectividad en la mitigación de amenazas.
- Diseñar un sistema web de detección temprana de ataques DoS en redes Wi-Fi, utilizando algoritmos de aprendizaje automático implementados en Python, que permita la clasificación y análisis del tráfico de red, asegurando la identificación y clasificación de anomalías relevantes.
- Implementar un proceso de validación del sistema web de detección temprana, que garantice su funcionalidad y capacidad de respuesta ante posibles ataques DoS, en entornos de red Wi-Fi, contribuyendo a la estabilidad y seguridad de la infraestructura de red frente a estos tipos de ataques.

El documento se estructura en tres capítulos: el primero abarca la revisión bibliográfica y conceptual que fundamenta esta investigación, proporcionando el marco teórico necesario para comprender los ataques DoS en redes Wi-Fi y el papel del aprendizaje automático en su detección. El segundo capítulo se centra en la descripción de las herramientas tecnológicas utilizadas, la metodología de desarrollo del proyecto, la configuración del entorno de pruebas. Finalmente, el tercer capítulo presenta los resultados obtenidos, la validación de la efectividad y la evaluación del impacto de la solución propuesta en la detección temprana de ataques DoS en redes Wi-Fi dentro de la Unidad de Tecnologías de la Información de la PUCE-I.

Capítulo I

Estado Del Arte

En este capítulo se expone los fundamentos teóricos y los conceptos esenciales que sustentan el desarrollo de un sistema web basado en aprendizaje automático, orientado a la detección temprana de ataques de denegación de servicio (DoS) en redes Wi-Fi. Se realiza una revisión detallada del funcionamiento de las redes inalámbricas, las principales amenazas de seguridad a las que se enfrenta los mecanismos existentes para mitigar los ataques DoS. Además, se analiza el papel del aprendizaje automático como una herramienta innovadora en la detección de intrusiones, considerando los algoritmos más empleados y tendencias actuales en el campo de la ciberseguridad. Este análisis busca proporcionar una base sólida para el diseño e implementación del sistema propuesto, en el contexto de redes Wi-Fi, dentro de este proceso formativo de la Pontificia Universidad Católica Del Ecuador Sede Ibarra.

1.1 Redes inalámbricas (Wi-Fi): fundamentos y contexto

1.1.1 Principios de las redes Wi-Fi.

Las redes Wi-Fi operan transmitiendo señales de radio en diferentes bandas de frecuencia, en lo principal 2.4 GHz y 5GHz, lo que permite ofrecer distintas velocidades y alcances de conexión. Por lo general, las frecuencias más altas, como la de 5GHz, proporcionan velocidades de transmisión superiores, pero la cobertura es mucho más limitada y tienen dificultad para atravesar obstáculos como paredes. Por otro lado, las bandas de 2.4 GHz, aunque ofrece velocidades más bajas, destaca por su mayor alcance y la mejor capacidad para penetrar obstáculos, lo ideal para cubrir áreas más grandes o los dispositivos ubicados lejos del router.

Las redes Wi-Fi tienen su funcionamiento como un sistema de comunicación en dos direcciones, entre lo que es el dispositivo y el router. Cuando un dispositivo se conecta, envía una solicitud de sondeo para así detectar las redes que tenga disponible en su proximidad. El router, también es conocido como un punto de acceso Wi-Fi, responde a las solicitudes de los dispositivos enviando un mensaje llamado baliza, que hace que contenga información como el nombre de la red (SSID), el tipo de cifrado y la intensidad de la señal disponible. (Proofpoint, 2024)

1.1.2 Arquitectura y componentes de una red inalámbrica.

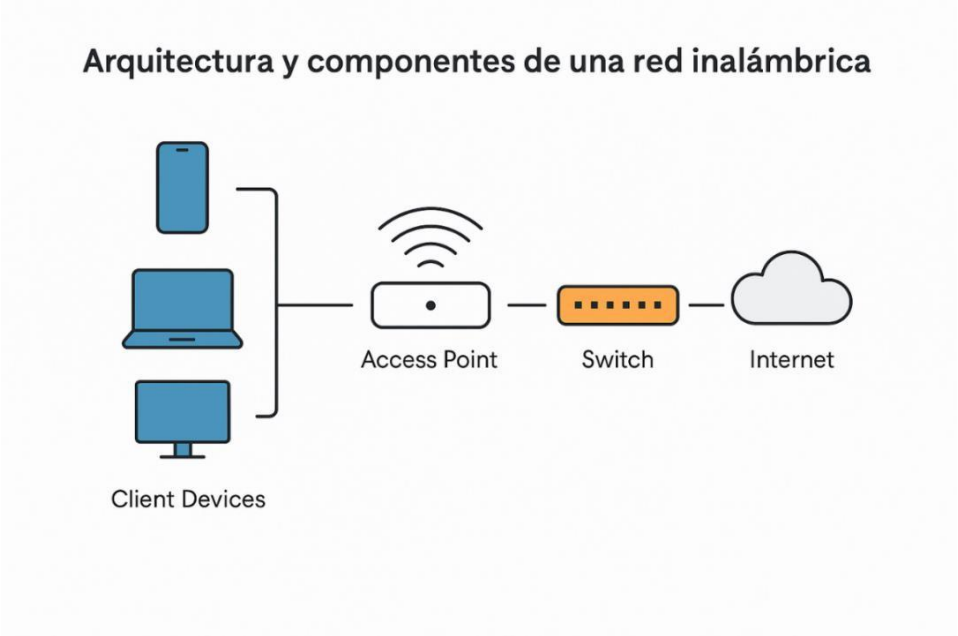
La arquitectura forma una base estructural de las redes inalámbricas, determinando cómo los dispositivos se conectan, comunican e interactúan entre sí. Esta arquitectura está formada por componentes muy importantes, como los puntos de acceso (AP), los dispositivos inalámbricos de los usuarios y la infraestructura de red para que pueda soportar.

Los puntos de acceso funcionan como puertas de enlace para la conexión de los dispositivos a la red, dando la posibilidad de envío y la recepción de datos de forma inalámbrica. La arquitectura de una red Wi-Fi comienza desde configuraciones simples, como un único punto de acceso en el hogar, hasta configuraciones en empresas con múltiples puntos de acceso, controladores y sistemas de gestión avanzados. Un diseño adecuado considera factores clave como la cobertura, la capacidad, la seguridad y la escalabilidad, con el objetivo de garantizar que la red sea robusta y eficiente, capaz de satisfacer las necesidades específicas de los usuarios y las aplicaciones. Este enfoque arquitectónico es fundamental para asegurar una conectividad estable y confiable. (Experts®, 2020)

En la ilustración 1, se muestra la arquitectura de una red inalámbrica, donde los dispositivos cliente se conectan a un punto de acceso, que a su vez se comunica con un switch para acceder a internet.

Ilustración 1

Arquitectura y componentes de una red inalámbrica



1.1.3 Desafíos y vulnerabilidades específicas de Wi-Fi.

Aunque las redes Wi-Fi ofrecen muchas conveniencias, también están sujetas a diversos riesgos y amenazas que pueden comprometer la seguridad. Una de las amenazas más frecuentes es el acceso no autorizado, donde atacantes aprovechan redes Wi-Fi sin la protección adecuada o con contraseñas muy débiles para conectarse sin permiso. Esto puede derivar en el robo, la interpretación de datos y la explotación de vulnerabilidades en los dispositivos conectados.

El phishing Wi-Fi es una estrategia muy utilizada por los ciberdelincuentes que consiste en crear redes inalámbricas falsas con nombres parecidos a las redes que se ocupa, tienen como objetivo engañar al usuario teniendo así información sensible. Los ataques de fuerza bruta representan una gran amenaza, estos atacantes intentan invadir el acceso no autorizado a redes probando sistemáticamente numerosas combinaciones de contraseñas débiles. (Noguera, 2023)

Las vulnerabilidades en las redes representan brechas de seguridad que los ciberdelincuentes pueden explotar para obtener acceso no autorizado. Esta práctica facilita enormemente la intrusión por parte de atacantes.

Las redes Wi-Fi son susceptibles a riesgos cuando el punto de acceso presenta configuraciones mal hechas. Por lo tanto, el no mantener actualizado el firmware del

punto de acceso puede permitir la explotación de fallas en seguridad, las cuales todos los fabricantes corrigen mediante actualizaciones regulares. (Noguera, 2023)

1.2 Seguridad en redes inalámbricas

1.2.1 Principales amenazas en entornos inalámbricos.

Una WLAN (Wireless Local Area Network) es una interconexión entre computadoras que utiliza tecnología inalámbrica, lo que permite la comunicación entre los dispositivos. Este tipo de interconexión se enlaza a través de un punto de acceso inalámbrico, que se encarga de emitir la señal de radio necesaria para establecer la conexión e interacción entre ellos.

Las redes de área local inalámbrica (WLAN) transmiten y reciben datos mediante ondas de radio, lo que elimina la barrera física proporcionada por los cables. Esta característica las hace vulnerables a la interpretación no autorizada, el espionaje, la piratería informática y otros riesgos de ciberseguridad.

Las amenazas a las redes WLAN y por tanto a las redes Wi-Fi, más comunes:

- Ataques de denegación de servicio: donde el intruso inunda la red con mensajes que afectan la disponibilidad de los recursos de red.
- Suplantación de identidad y secuestro de sesión: donde el atacante obtiene acceso a datos y recursos de la red asumiendo la identidad de un usuario válido.

Para mitigar estas amenazas, es crucial configurar adecuadamente su red WLAN y activar diversas funciones de seguridad, incluyendo la autenticación y el cifrado estándar, así como otros mecanismos de control de acceso. (nibusinessinfo.co.uk, s.f.)

1.2.2 Técnicas de protección y prevención.

La protección en redes inalámbricas es primordial para garantizar la integridad, confidencialidad y disponibilidad de los datos. Debido a las señales inalámbricas propagadas por el aire, están más expuestas a ataques que las redes cableadas. Por ello se han desarrollado técnicas de protección y prevención que permiten reducir los riesgos asociados al uso de estas tecnologías.

- Cifrado de la red
- Contraseñas Robustas
- Filtrado de direcciones MAC

- Ocultamiento del SSID
- Actualización del firmware
- Desactivación de funciones innecesarias
- Monitoreo de la red

1.2.3 Importancia de la detección temprana de ataques.

La detección temprana de ataques es fundamental para la seguridad de las redes. Ante un entorno digital cada vez tiene a ser vulnerable, identificando actividades maliciosas en sus primeras etapas permite minimizar daños, responder con rapidez y prevenir la propagación de amenazas dentro del sistema.

- Reducción del impacto
- Prevención de pérdidas económicas
- Protección de la integridad y privacidad de los datos
- Reacción inmediata ante amenazas
- Mejora continua de la seguridad
- Cumplimiento normativo y legal

1.2.4 Desafíos actuales en la seguridad de redes WiFi.

En la actualidad, las redes Wi-Fi son esenciales, ya sea para mantenerse en contacto con familiares y amigos o para realizar reuniones de trabajo, cambiando de manera significativa la manera en que las personas se logran comunicar. No obstante, al igual que cualquier tecnología, las redes Wi-Fi enfrentan desafíos inherentes. (ACT Fibernet, 2024) como, por ejemplo:

Interferencias: Uno de los problemas más frecuentes en las redes Wi-Fi son las interferencias, que pueden ser causadas por otros dispositivos electrónicos como microondas, teléfonos inalámbricos o redes cercanas

Alcance: Otro problema muy común es su alcance. Obstáculos como paredes, suelos y otros elementos pueden bloquear la señal, lo que conduce a conexiones débiles o inexistentes en determinadas áreas del edificio.

Congestión: En áreas con alta intensidad de población, las redes Wi-Fi pueden congestionarse, especialmente durante las horas de mayor demanda. Esto puede resultar

en velocidades más lentas y una mayor latencia, lo que complica la transmisión de videos o la descarga de archivos de gran tamaño.

1.3 Ataques de denegación de servicio

1.3.1 Definición, objetivos y características.

Un ataque de denegación de servicio (DoS) es un ciberataque en el que los atacantes intentan hacer que el sistema, servidor o dispositivo deje de estar disponible para sus usuarios. Esto logra sobrecargarlo con un volumen excesivo de tráfico o solicitudes maliciosas, agotando sus recursos para provocar la interrupción del funcionamiento.

Características de un ataque de Denegación de Servicio (DoS)

- Saturación de recursos
- Interrupción del servicio
- No busca robar datos
- Duración variable
- Fácil de detectar, difícil de mitigar
- Afecta la disponibilidad
- Puede lanzarse contra diferentes objetivos

1.3.2 Tipología: DoS Y DDoS (ataques distribuidos).

Ataque de Denegación de Servicio

Un DoS se trata de un ataque malintencionado cuyo objetivo es bajar el rendimiento del servidor o impedir su disponibilidad. Este tipo de ataque satura el recurso atacando con un volumen excesivo de solicitudes, lo que agota los recursos del sistema causando una interrupción en el servicio. (Radware, s.f.)

El propósito de un ataque DoS es impedir que un sistema o recursos de red estén accesibles para los usuarios legítimos, ya sea al bloquear el sistema o saturar su ancho de banda lo que provoca que deje de funcionar correctamente.

En la ilustración 2, se presenta una representación gráfica de un ataque DoS, en la cual un único equipo atacante envía solicitudes de manera repetitiva a un servidor con el objetivo de sobrecargar sus recursos y afectar su disponibilidad.

Ilustración 2

Representación gráfica de un ataque DoS

Nota: Imagen tomada de (ManageEngine, 2025)

Tipos de Ataques DoS

1. **Ataques de Denegación de Servicio Basados en Volumen:** Estos ataques intentan saturar el ancho de banda de la víctima mediante gran volumen de tráfico. (S. Abiramasundari y V. Ramaswamy, 2025)
2. **Ataques a Nivel de Aplicación:** Incluye técnicas como el HTTP Flood, en donde se envían un gran número de solicitudes HTTP para agotar los recursos del servidor. (Sharma, 2023)
3. **Ataques de Reflejo y Amplificación:** Utilizan servicios vulnerables para enviar tráfico masivo a la víctima. Los ataques UDP y DNS son comunes en esta categoría. (Ouhssini, 2024)
4. **Ataques Distribuidos (DDoS):** Implican múltiples dispositivos botnet que atacan simultáneamente. (Almadhor, 2024)

Métodos de defensa contra ataques de denegación de servicio

Las maniobras de prevención buscan impedir que un ataque se lleve a cabo antes de que suceda realmente. Estas tácticas permiten realizar ajustes en protocolos, aplicaciones y sistemas para aumentar su capacidad de resistir intentos de ataque. En el caso de los ataques DoS, la prevención se enfoca en disminuir la probabilidad de ser vulnerables y dificultar que el atacante reúna una gran cantidad de agentes, lo que esto reduce a los ataques tengan éxito. (D. Narváez, 2020) como, por ejemplo:

Algunos métodos de Defensa de Ataques DoS son:

- **Método de Seguridad del Sistema:** Se enfocan en prevenir el acceso no autorizado al sistema en general.
- **Método de Seguridad en Protocolo:** Abordar vulnerabilidades inherentes al diseño de los protocolos de comunicación.
- **Métodos de Supervisión de Recursos:** Controlar y regular el acceso de los usuarios a los recursos del sistema.
- **Métodos de Multiplicación de Recursos:** Buscar la replicación o distribución de recursos para mejorar la disponibilidad, el rendimiento o la tolerancia a fallos.

Ataque de Denegación de Servicio Distribuido

Un ataque DDoS es una acción maliciosa destinada a interrumpir el funcionamiento de la red o servidor mediante la saturación, con un gran volumen de solicitudes innecesarias que provienen de múltiples fuentes. La sobrecarga bloquea la capacidad del sistema para gestionar el tráfico legítimo y responder correctamente a las peticiones. La diferencia que tiene de los ataques DoS es que provienen de una sola fuente, en el caso de los ataques DDoS ocupa varios dispositivos infectados, más conocidos como botnet, ubicados en varios lugares. (Radware, s.f.)

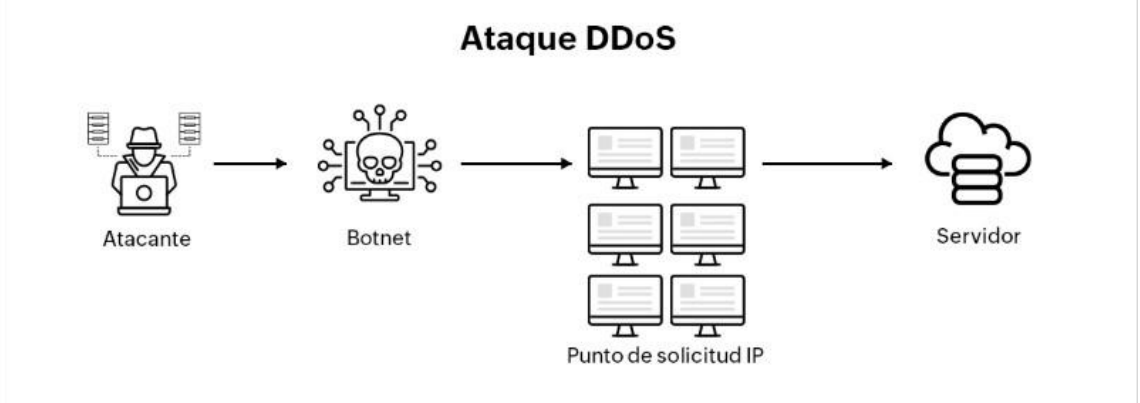
Los ataques de denegación de servicio distribuidos suelen experimentar un crecimiento enorme, independientemente de la forma en las que sean medidos. La frecuencia de los incidentes, el ancho de banda máximo medio y la complejidad de los ataques están aumentando, y parece que ninguna organización está a salvo.

Los ataques de Denegación de Servicio Distribuida (DDoS) son una de las amenazas cibernéticas más comunes y peligrosas en la actualidad. Consisten en sobrecargar un servidor, red o servicio con un gran volumen de tráfico malicioso.

Como se muestra a continuación en la ilustración 3, se presenta un diagrama representativo de un ataque DDoS distribuido, donde múltiples dispositivos coordinados generan tráfico malicioso hacia un único objetivo, con el fin de saturar sus recursos y provocar su inaccesibilidad.

Ilustración 3

Diagrama representativo de un ataque DDoS distribuido



Nota: Imagen tomada de (ManageEngine, 2025)

Impacto en la disponibilidad del sistema y calidad del servicio.

El efecto más inmediato de un ataque DoS es la pérdida del sistema, provocando periodos de inactividad o la interrupción completa del servicio. Esto puede derivar en la pérdida de conexión, la caída de servidores o la paralización de aplicaciones esenciales.

La calidad del servicio (QoS) hace referencia a la habilidad de una red o sistema para ofrecer un rendimiento óptimo en términos de velocidad, latencia y confiabilidad. Sin embargo, un ataque DoS puede afectar negativamente a los aspectos de QoS, y más en servicios que requieren respuestas rápidas.

1.4 Aprendizaje automático

El aprendizaje automático (Machine Learning) es fundamental en el campo de la inteligencia artificial, que se enfoca en el desarrollo de algoritmos y modelos capaces de aprender, extraer patrones a partir de datos, sin requerir una programación explícita para cada tarea específica. Su objetivo principal radica en mejorar el desempeño en tareas concretas mediante la experiencia adquirida, lo que facilita la identificación de tendencias, la realización de predicciones y la toma de decisiones de manera autónoma. Esta capacidad de aprendizaje y adaptación convierte al aprendizaje automático en una herramienta esencial para el diseño de soluciones innovadoras, contribuyendo de manera significativa al avance tecnológico.

Para aprovechar al máximo el aprendizaje automático, es fundamental proporcionarle datos precisos y de alta calidad. Considera que el aprendizaje automático solo puede

aprender de datos que se le proporciona, por lo que, si los datos son incorrectos, los conocimientos derivados tambien serán erróneos. (Maciej Bartłomiej Sikora, 2024)

1.4.1 Tipos de aprendizaje: supervisado, no supervisado.

Modelo de aprendizaje automático supervisado: El aprendizaje supervisado es un método dentro del aprendizaje automático que se emplea para entrenar sistemas informáticos con el propósito de obtener predicciones exactas. Este método utiliza un conjunto de datos con etiquetas, donde cada instancia cuenta con una etiqueta que señala el resultado correcto.

Modelo de aprendizaje automático no supervisado: El aprendizaje no supervisado es un enfoque, que utiliza datos no etiquetados para entrenar un modelo.

A diferencia del aprendizaje supervisado, esta técnica no cuenta con un conjunto de datos previamente etiquetados para guiar el entrenamiento, en su lugar, el modelo debe encontrar por sí mismo patrones y estructuras dentro de los datos.

1.4.2 Ventajas del uso de aprendizaje automático frente a sistemas tradicionales de detección

El aprendizaje automático presenta ventajas significativas frente a los métodos tradicionales para la detección de ataques DoS en redes Wi-Fi. Entre los beneficios se destacan la capacidad de procesar grandes volúmenes de datos en tiempo real, identificar patrones atípicos que podrían pasar desapercibidos mediante técnicas convencionales, adaptarse automáticamente a nuevas amenazas y mejorar continuamente la eficacia en la detección mediante el uso de herramientas inteligentes. (Peral, 2019)

1.5 Algoritmos de aprendizaje automático aplicados a la detección de DoS.

El uso de los algoritmos de aprendizaje automático hoy en día se ha convertido en una gran estrategia para la detección de ataques DoS, ya que se tiene por analizar grandes volúmenes de tráfico de red y detectar comportamientos anómalos. Lo que permiten estos algoritmos es el proceso de identificación de amenazas, reduciendo el tiempo de respuesta y poder mejorar la seguridad en las redes.

A continuación, se presentan los algoritmos más utilizados

- **Random Forest:** Es un bosque aleatorio de árboles de decisión que combina múltiples árboles independientes (mediante bagging y aleatoriedad de características) para ofrecer gran robustez y resistencia al sobreajuste.
- **Máquinas de Vectores de soporte (SVM):** Se caracterizan por encontrar el hiperplano óptimo de separación que maximiza el margen entre las clases en un espacio de alta dimensión, generalizando bien.
- **XGBoostClassifier:** Es un algoritmo de Gradient Boosting altamente optimizado que construye árboles de decisión secuencialmente, corrigiendo los errores residuales de los modelos anteriores para lograr una precisión superior.
- **LightGBMClassifier:** Su característica principal es su eficiencia y velocidad. Es un algoritmo de Gradient Boosting, pero utiliza técnicas innovadoras para reducir el tiempo de entrenamiento y el consumo de memoria en grandes datasets sin sacrificar precisión.

La Tabla 1 muestra la comparación que existe entre los algoritmos de aprendizaje automático Random Forest, Máquinas de Vectores de Soporte, LightGBMClass, XGBoost, buscando aspectos como precisión, velocidad de entrenamiento. Es fundamental para poder seleccionar el algoritmo más adecuado para realizar la detección de ataques DoS.

Tabla 1

Comparación de Algoritmos de Aprendizaje Automático

Característica	**Random Forest (Wikipedia contributors, 2025)**	**SVM (Máquinas de Vectores de Soporte) (Wikipedia contributors, 2025)**	**LightGBMClassifier** (Team), 2025)	**XGBoost (XGBoostClassifier)** (IBM Developer Team, 2023)
Tipo de algoritmo	Ensamble (basado en árboles)	Clasificación binaria/multiclase	Ensamble (Gradient Boosting)	Ensamble (Gradient Boosting)
Precisión	Alta, especialmente con datos ruidosos	Muy alta en problemas bien definidos	Muy alta, eficiente con grandes datos	Alta
Velocidad de entrenamiento	Rápido a moderado	Lento con grandes volúmenes de datos	Muy rápida	Rápido y Optimizado

Interpretabilidad	Alta (puede explicarse con árboles)	Media (difícil visualizar los márgenes)	Media (difícil interpretar árboles complejos)	Media (difícil interpretar árboles)
Manejo de ruido y datos faltantes	Muy bueno	Sensible al ruido	Muy bueno (maneja valores faltantes)	Muy bueno (maneja valores faltantes)
Necesidad de ajuste de parámetros	Baja	Alta (kernel, C, gamma)	Moderada-Alta (varios para optimizar)	Alta (muchos parámetros para optimizar)
Uso común	Clasificación, regresión, detección de anomalías	Clasificación de texto, imágenes, bioinformática	Clasificación regresión (grandes datasets)	Clasificación, regresión
Ventaja principal	Robusto y fácil de implementar	Precisión y buen manejo de margen	Velocidad y eficiencia con alta precisión	Rendimiento superior, velocidad
Desventaja principal	Puede ser costoso en memoria	Dificultad para escalar a datos muy grandes	Potencial de sobreajuste si no se ajusta bien	Propenso a sobreajuste si no se ajusta bien.

Nota: Tabla creada por Diego Pabón

1.5.1 Proceso de entrenamiento y validación.

El desarrollo de un sistema web basado en aprendizaje automático para detectar ataques DoS en redes Wi-Fi también implica hacer un proceso exhaustivo de entrenamiento y validación de los modelos. El proceso es crucial para garantizar que los algoritmos puedan reconocer patrones maliciosos en el tráfico de red con precisión y que tengan la capacidad de generalización

1.5.2 Métricas de evaluación.

La evaluación del rendimiento de los modelos resulta fundamental para determinar su capacidad predictiva y la efectividad en escenarios reales. En particular, para la detección de ataques DoS, donde los datos suelen presentar un marcado desequilibrio con mayor cantidad de tráfico legítimo que malicioso, es imprescindible emplear una variedad adecuada de métricas que permitan medir con precisión el desempeño del modelo. Estas

métricas facilitan una evaluación integral que considera tanto la identificación correcta de ataques como la minimización de falsos positivos para garantizar la fiabilidad y utilidad del sistema en entornos prácticos.

1.6 Estándares y protocolos (IEEE 802.11, WPA, WPA2, etc.)

Protocolo IEEE 802.11

Creado en 1997 por el Institute of Electrical and Electronics Engineers (IEEE), es un conjunto de estándares que busca regular la comunicación inalámbrica en redes de área local (WLAN) y ha evolucionado para ofrecer métodos eficientes y confiables que minimizan colisiones y aseguran una transmisión segura de los datos. Este estándar establece las especificaciones para la capa física y la subcapa de acceso, teniendo la facilidad de interacción entre los diferentes dispositivos. Además, incluye diversas categorías de redes inalámbricas y ha incorporado mejoras como calidad de servicio, seguridad y movilidad eficiente, adaptándolas a distintas aplicaciones y entornos. Así, IEEE 802.11 facilita una conexión inalámbrica estable, segura y compatible entre dispositivos, independientemente del fabricante o modelo siendo la base tecnológica más utilizada para las redes.

Este protocolo incluye mecanismos de seguridad para proteger las redes inalámbricas contra posibles ataques que abarcan la encriptación de datos y la autenticación de dispositivos, lo cual es vital debido a que la transmisión de datos sea susceptible a ser interceptada por personas con el equipo adecuado. Por lo tanto, la seguridad en las redes inalámbricas es muy importante para poder garantizar la integridad y la confidencialidad de la información transmitida.

WPA (Wi-Fi Protected Access)

El protocolo WPA fue introducido como una solución provisional para poder mejorar la seguridad frente a las vulnerabilidades del protocolo WEP. Aunque WPA utiliza algoritmo TIKP basado en RC4, utilizando una clave de 128 bits e inicializando con un vector de 48 bits, no emplea directamente una clave de 256 bits como se ha mencionado. Sin embargo, WPA representa un avance considerable respecto a WEP, ya que esto incorpora un cifrado más robusto con métodos de autenticación mejorados, lo que lo hace más seguro que su antecesor, que usaba claves de 64 o 128 bits. (Software, 2021)

WEP (Wired Equivalent Privacy)

Wilder Equivalent Privacy (WEP) es un protocolo de seguridad para redes inalámbricas introducido en 1997 con el objetivo de ofrecer un nivel de protección similar al de una red cableada. WEP cifra los datos transmitidos a través de la red Wi-Fi usando una clave compartida estática de 64 o 128 bits, que los dispositivos autorizados puedan enviar y recibir información de forma segura. (Staff, 2024)

WPA2 (Wi-Fi Protected Access 2)

El WPA2, segunda generación del protocolo de seguridad inalámbrica Acceso Wi-Fi Protegido, fue creado para asegurar las redes Wi-Fi, protegiendo la seguridad transmitida esté cifrada para que sea accesible para quienes cuentan con la contraseña. Una de las principales ventajas de WPA2 es que el uso cifrado avanzado AES, que reemplaza al menos seguro TKIP utilizando WPA. El AES, es un algoritmo de cifrado robusto, que incluso es utilizado en el gobierno de Estados Unidos para poder proteger la información clasificada. (Software, 2021)

1.7 Antecedentes

Diversos estudios han evidenciado el reciente crecimiento y riesgo que se enfrentan las micro, pequeñas y medianas empresas (MIPYMES), en Latinoamérica frente a las amenazas cibernéticas. Según un 25% de las pymes en la región ha sido víctima de ciberataques, entre ellos los ataques de denegación de servicios (DoS), los cuales pueden manejar por completo la operabilidad de los sistemas empresariales. Esta situación es especialmente crítica considerando que muchas de las microempresas no tienen recursos técnicos ni económicos suficientes para poder implementar medidas de seguridad robusta. (Manapro Consultores, s.f.)

(Rodriguez, 2021) este artículo, explora cómo los estudiantes utilizan las plataformas sobre los riesgos asociados con la divulgación de información. Hay un estudio exhaustivo que permite analizar en profundidad los factores que contribuyen al manejo inadecuado de la información y a la carencia de responsabilidad entre los empleados. El estudio examina los procesos internos y las prácticas organizacionales que favorecen estas deficiencias, identificando tanto las causas estructurales como las conductas individuales que inciden negativamente en la gestión de la información y en el cumplimiento de las responsabilidades asignadas.

(Mero, Ortiz, Yanangómez, & Rodríguez, 2021) habían realizado un análisis de seguridad de redes inalámbricas en el campo de tecnologías de la información con el fin de identificar protocolos y métodos de seguridad utilizados diferentes métodos. Las nuevas innovaciones no sólo refuerzan la protección contra los ciberdelincuentes, sino también que pueden mejorar la integridad para que se vuelvan más seguras y confiables.

El aprendizaje automático se ha consolidado como una gran herramienta fundamental para proteger a las redes Wi-Fi frente a ataques complejos. Estudios recientes indican que métodos como redes neuronales profundas y máquinas de soporte vectorial permiten con una gran precisión detectar el filtro de los ataques. Como, por ejemplo, (Sreeram, A., & Saravanan, K., 2020) desarrollaron un sistema de detección de intrusos basado en aprendizaje automático que fue XGBoost logrando una precisión de 99,42% al poder clasificar registros de tráfico en categorías como normal, suplantación o inyección.

En el trabajo realizado por (Cuvi, 2023) se evaluó la confiabilidad de que los estudiantes y docentes de la Pontificia Universidad Católica del Ecuador puedan conectarse a redes Wi-Fi abiertas, destacando las vulnerabilidades y riesgos asociados, como la pérdida de información y suplantación de identidad. Como resultado, se resaltó la necesidad de concienciar a los estudiantes sobre la importancia de adoptar prácticas seguras al acceder a internet.

(Jurado, Escobar, & Carrión, 2021) destacan que muchas de las microempresas en Ecuador operan sin protocolos adecuados de seguridad informática y sin infraestructura tecnológica capaz de detectar o mitigar amenazas como lo que son ataques DoS. Esta falta de preparación técnica convierte a estas organizaciones en puntos fáciles para los ciberdelincuentes.

En Ecuador, diversas instituciones académicas han abordado la implementación de técnicas de aprendizaje automático y agentes inteligentes dentro del marco de la inteligencia artificial para optimizar la seguridad en redes inalámbricas. Por ejemplo, en la Universidad Técnica de Machala, se desarrolló un sistema basado en agentes inteligentes que trabajan de manera autónoma para detectar amenazas de phishing en redes de área local (LAN). Este sistema empleó algoritmos de clasificación, específicamente sospechosos, Además, se utilizó el entorno simulado GNS3 y la metodología CRISP-DM, estructurando un proceso sistemático para la construcción del modelo. Los agentes se encargaron de monitorear continuamente la red, identificar

patrones anómalos y alertar en tiempo real posibles amenazas, logrando así un sistema de seguridad inteligente eficiente y autónomo. (Morales Tapia, 2023)

Capítulo II

Materiales y Métodos

En este capítulo se exponen los aspectos generales del proyecto, describiendo así las etapas de planificación y desarrollo de la metodología propuesta, diseñar un sistema web basado en aprendizaje automático, que está enfocado a la detección temprana de ataques DoS en redes Wi-Fi. Asimismo, el enfoque adoptado para la recolección y tratamiento de datos, como también las consideraciones técnicas necesarias para la conceptualización de un entorno de análisis eficiente.

2.1 Aspectos generales de la Investigación

La investigación se enmarca dentro del tipo aplicado porque se utilizan los conocimientos adquiridos durante la carrera, desarrollándose en el contexto del uso de tecnologías emergentes. Se abordó un aspecto crucial para el campo de la seguridad, en el diseño conceptual de un sistema web basado en técnicas de aprendizaje automático para la detección temprana de ataques de denegación de servicio (DoS) en redes Wi-Fi, que va dirigido a la Unidad de Tecnologías de la Información de la Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCE-I).

Se adoptó un enfoque cuantitativo que permitió analizar datos provenientes de un Dataset, obtenidos mediante pruebas en un entorno controlado, simulaciones y la evaluación de métricas clave relacionadas con el comportamiento de la red frente a posibles ataques DoS. Este enfoque facilitó una evaluación objetiva del sistema propuesto, así como la mediación precisa de la efectividad de los algoritmos de aprendizaje automático en la detección de las amenazas.

2.2 Metodología de la propuesta

Procedimiento de la Investigación

Para poder llevar a cabo el desarrollo del sistema web basado en aprendizaje automático para la detección temprana de ataques DoS en redes Wi-Fi, este proceso se enfocó en un análisis, para asegurar la calidad de los resultados, se validaron los datos usando métricas como precisión, recall y F1-Score.

2.2.1 Fase I: Modelo de Aprendizaje Automático

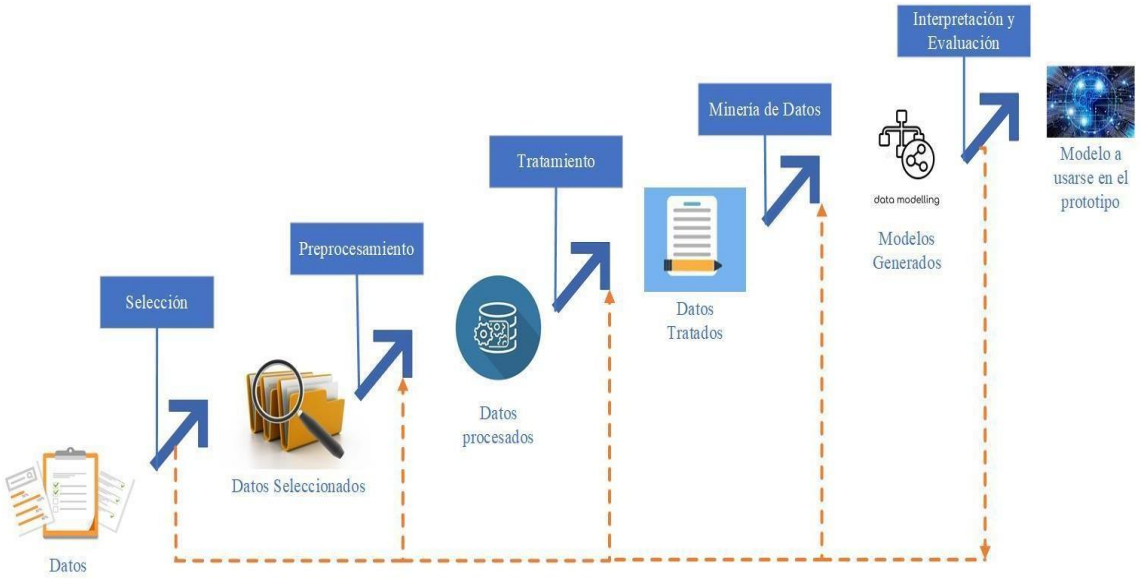
Para organizar el proceso y el conocimiento obtenidos a través de los datos de tráfico de red, se utilizó la metodología KDD (Descubrimiento de Conocimiento en Bases de Datos), que proporciona un enfoque sistemático para la extracción de conocimiento a partir de grandes volúmenes de datos. (Nick Hotz, 2024)

La implementación de esta fase incluye cinco etapas principales: selección de datos, preprocesamiento, transformación, modelado y evaluación. Con el fin de construir un modelo eficaz que sea capaz de detectar el tráfico de red bajo el estándar IEEE 802.11.

A continuación, en la ilustración 5 se muestra el proceso que se llevó a cabo:

Ilustración 4

Fases del método de extracción de conocimiento KDD de datos (aprendizaje automático)



Nota: Imagen tomada de la tesis Prototipo de un sistema predictivo para la deserción estudiantil en el contexto universitario, por Alexander Ortiz.

Selección de datos

La selección de datos constituyó el primer paso fundamental en el desarrollo del sistema. Para ello, se llevó a cabo un análisis detallado de las especificaciones técnicas del estándar IEE 802.11, con el objetivo de identificar aquellas características del tráfico que puedan servir como indicadores de posibles ataques de denegación de servicio.

El estándar IEEE 802.11 define el comportamiento de las tramas de gestión, control y datos, tales como los mecanismos de acceso al medio, retransmisión, control de gestión

y manejo de errores. Los ataques DoS contra redes Wi-Fi suelen explotar debilidades en estos mecanismos, ya sea saturando el canal con tráfico inútil, forzando desconexiones repetidas o sobrecargando los puntos de acceso.

A partir de este análisis, se seleccionaron las siguientes variables como atributos de entrada para el modelo como se especifica en la tabla 2:

Tabla 2

Variables

Variable	Descripción	Justificación
flow_duration	Duración de la sesión de red	Las conexiones legítimas suelen mantenerse durante varios segundos, mientras que los ataques SYN flood ocurren en menos de 0.1 segundos.
total_fwd_packets	Número total de paquetes transmitidos	Se envía entre 1 y 10 paquees, mientras que un atacante puede enviar más de 100 paquetes en un período de 5 segundos para saturar la red.
flow_packets_per_second	Velocidad de envío de paquetes por segundo	Se hacen 50 paquetes por segundo, mientras que las herramientas automáticas superan los 1,000.
total_backward_packets	Número de respuestas del servidor	Se generan paquetes de tamaños variados, mientras que los programas automatizados suelen enviar paquetes idénticos, durante un período de 3 segundos, se puede

		detectar actividad automatizada maliciosa.
fwd_packet_length_std	Variación en el tamaño de los paquetes	Se tiende a enviar paquetes con tamaños variados, mientras que los programas generan paquetes uniformes, durante un intervalo de 3 segundos para la presencia maliciosa.
total_backward_packets	Número de respuestas del servidor	Los servidores responden correctamente, pero cuando están saturados no pueden hacerlo.
protocol	Tipo de comunicación (TCP/UDP)	El tráfico habitual esta por 60%, el UDP puede superar el 80%.
dst_port	Puerto de destino del ataque	El tráfico normal usa muchos puertos, pero los ataques se enfocan en puertos específicos.
total_length_fwd_packets	Bytes totales enviados hacia el objetivo	Los ataques SYN usan paquetes muy pequeños y los UDP usan paquetes muy grandes. Detectar estos tamaños extremos en 3 segundos indica un ataque.

Estas características fueron seleccionadas por su alta relación con el comportamiento anómalo en redes inalámbricas y su viabilidad de extracción a través de herramientas de monitoreo como Wireshark, la cual permite capturar tráfico.

Preprocesamiento de datos

Los archivos de tráfico de red en formato. pcap se convierten a archivos .csv utilizando herramientas como tshark o librerías especializadas de Python. Se realizaron depuraciones de los datos para eliminar registros incompletos, duplicados o que no aportan información relevante.

Se aplicaron los siguientes pasos de limpieza:

- **Eliminación de registros incompletos o nulos:** Se suprimieron filas que los valores faltantes o inconsistentes en cualquiera de los atributos seleccionados, asegurando la integridad de cada instancia de datos.
- **Eliminación de duplicados:** Se identificaron y eliminaron registros idénticos para evitar sesgos durante el entrenamiento del modelo y garantizar que cada observación fuera única.
- **Conversión de tipos de datos:** Todos los atributos fueron convertidos a tipos de datos numéricos (entero y flotantes) apropiados para los algoritmos de aprendizaje automático, facilitando su procesamiento.
- **Filtrado de columnas,** conservando únicamente los atributos definidos: Duration, Packets, bytes, src_port, dst_port, rate.

El conjunto de datos resultante fue etiquetado de forma supervisada, asignando clases a cada instancia con base en el escenario simulado:

- **0:** tráfico normal
- **1:** tráfico malicioso

Transformación de datos

La transformación de los datos consistió en preparar los atributos seleccionados para entrenar al modelo.

- Se aplicó el escalado de características utilizando el método *StandardScaler* de la biblioteca Scikit-learn. Este método fue fundamental para los modelos sensibles a la magnitud de los datos, como *Máquinas de Vectores (SVM)*, estandarizando los atributos para que tuvieran una media de 0 y una desviación de 1. Permitiendo que el modelo aprenda de manera equilibrada
- No fue necesario la codificación de variables categóricas, ya que todos los atributos fueron numéricos.

Minería de datos (aprendizaje automático)

Para la presente propuesta se utilizó el conjunto de datos CICDDoS2019, desarrollado por el Canadian Institute for *CyberSecurity*, está etiquetado, ya que cada muestra incluye una columna *label* que indica si el tráfico es normal o corresponde a un ataque DoS o DDoS, permitiendo un entrenamiento supervisado. Este Dataset contiene diversas características del tráfico de red, para el entrenamiento del modelo se utilizó todas las variables disponibles en el conjunto de datos, sin realizar selección adicional, con el fin de aprovechar toda la información que ofrece para mejorar la detección de ataques. ((CIC), 2019).

Los archivos del Dataset fueron utilizados directamente en su versión en formato Parquet, un formato columnar optimizando para el almacenamiento eficiente y carga rápida de grandes volúmenes de datos. Este formato permitió archivos, tales como UDPLag.parquet, MSSQL.parquet, LDAP.parquet, entre otros, los cuales contienen registros previamente etiquetados y procesados. ((CIC), 2019).

La elección del Dataset CICDDoS2019 en formato Parquet responde a criterios de calidad técnica como la eficiencia operativa, ya que permite trabajar con datos altamente estructurados y optimizados en tiempos de lectura y análisis, facilitando su integración con el modelo de aprendizaje. La base de datos consta de aproximadamente millones de registros distribuidos en múltiples columnas que describen diferentes aspectos del tráfico de red, como duración, tamaño de paquetes, Flags, tasas y *subflows*, entre otros, y también de columnas. La estructura incluye características numéricas, categorías y de tiempo, además de una columna de etiquetas llamada *label*, que indica si la muestra corresponde a tráfico normal o diferentes ataques. El Dataset abarca variedad de ataques, volumen muy representativo lo que convierte en una base sólida para evaluar el desempeño de los algoritmos.

Entrenamiento del modelo

Una vez que se dispone del Dataset cargado y procesado desde archivos en formato Parquet, se procede a las fases finales previas al entrenamiento del modelo. A continuación, se llevó a cabo el entrenamiento de varios modelos supervisados utilizando la misma biblioteca, evaluando algoritmos como: Random Forest, XGBoostClassifier, Support Vector Machine y LightGBM. El objetivo principal fue identificar el modelo más eficaz para clasificar correctamente el tráfico como normal o malicioso.

El proceso seguido para entrenar y seleccionar el modelo más adecuado se estructuró en las siguientes etapas:

Etapas del proceso:

Etapa 1: División de datos

El conjunto de datos fue divido entre 80% entrenamiento y 20% validación, utilizando una validación cruzada de k pliegues (k -fold cross-validation), con el fin de garantizar una evaluación más robusta y generalizable. La validación cruzada se seleccionó porque permitió que todos los datos se utilicen tanto para el entrenamiento como para la evaluación del modelo, lo que proporciona una estimación más confiable del rendimiento. Esto resulto relevante en la detección de ataques DoS, donde resulta fundamental contar con una evaluación rigurosa y alta confianza en los resultados obtenidos. El proceso asegura que los datos utilizados para evaluar el rendimiento del modelo no han sido utilizados durante el entrenamiento, lo que proporciona una estimación más realista de su efectividad en datos nuevos.

Etapa 2: Escalado

Esta fase forma parte del proceso de preprocesamiento de los datos. Para los modelos que son sensibles a la escala de las variables, se utilizó *StandardScaler* para normalizar las características en el conjunto de entrenamiento, restando la media y dividiendo entre la desviación estándar. Luego, se aplicó la misma transformación al conjunto de validación, asegurando que las escalas de las variables sean consistentes y evitando fugas de información entre los conjuntos.

Etapa 3: Entrenamiento

Para el entrenamiento del modelo se utilizó cuatro arquitecturas de aprendizaje automático supervisado: Random Forest, que consiste en un conjunto de árboles de decisión, XGBoostClassifier, un modelo Gradient Boosting que construye árboles de manera secuencial, corrigiendo los errores iteraciones; SVM que encuentra el hiperplano que maximiza la separación entre las diferentes clases; y LightGBM, un algoritmo de Gradient Boosting optimizado que crece los árboles hoja por hoja, mejorando la eficiencia del entrenamiento.

Etapa 4: Evaluación

Se evaluó cada modelo con las métricas

- Precision (accuracy): Proporción de predicciones correctas sobre el total de predicciones.
- Precision positiva (precisión): Proporción de verdaderos positivos sobre el total de predicciones positivas, cuantos de los ataques fueron detectados.
- Recall (tasa de verdaderos positivos): Proporción de verdades positivos sobre el total de casos positivos reales.
- F1-score: La media armónica de precisión y recall, proporcionando un equilibrio entre ambas métricas.

Etapa 5: Selección

Se seleccionó el modelo que presentó el mejor rendimiento, alcanzando valores elevados en las métricas evaluadas y demostrando un equilibrio óptimo entre precisión y sensibilidad para la detección de ataques DoS. La elección se fundamentó en la capacidad del modelo para clasificar el tráfico con alta fiabilidad y consistencia, garantizando así un desempeño robusto y confiable en escenarios prácticos.

Fase 6: Exportación

El modelo fue guardado como. pkl utilizando joblib. Esto permitió la carga y el uso en el sistema web de manera eficiente, sin necesidad de realizar reentrenamiento.

Evaluación

Para generar la evaluación cuantitativa, se generó una matriz de confusión con ConfusionMatrixDisplay de Scikit-learn, la cual fue visualizada usando matplotlib. Esta matriz permitió un análisis detallado de los verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos.

Además, se representó la proporción de tráfico normal vs malicioso mediante gráficos de pastel. Estas representaciones visuales permiten validar fácilmente la efectividad del modelo

2.3 Fase II: Sistema Web

La segunda fase del proyecto fue la implementación del sistema web que integra el modelo de aprendizaje automático previamente entrenado para la detección temprana de ataques DoS en redes Wi-Fi. Para el desarrollo de esta etapa se aplicó la metodología XP, que es un enfoque ágil que nos entrega funcionalidades, adaptables al cambio y mejoras.

Aplicación de la Metodología XP

Durante esta fase, se aplicaron los principios de Extreme Programming (XP), tales como el desarrollo incremental, la comunicación continua, el diseño evolutivo y codificación enfocada en la simplicidad. Las funcionalidades clave del sistema fueron organizadas en historias de usuario que guiaron cada etapa del desarrollo. En la Tabla 3 se presentan las historias de usuario consideradas necesarias para el proyecto.

Niveles de Importancia

Critica: Indispensable. 90-100

Alta: Muy importante. 60-89

Media: Importante 59-50

Baja: No prioritaria.

Tabla 3

Historias de Usuario

ID	NOMBRE	ESTIMACION ESFUERZO (HORAS)	IMPORTANCIA	DESCRIPCIÓN DE HISTORIA DE USUARIO	CRITERIOS DE ACEPTACIÓN	DEPENDENCIAS
HU-01	Carga y Clasificación de Tráfico	40	Alta	Como analista de ciberseguridad, quiero cargar archivos CSV generados desde Wireshark para que el sistema analice y clasifique el tráfico automáticamente.	● La interfaz debe permitir al usuario seleccionar y subir un archivo .csv desde su ordenador. ● El sistema debe leer correctamente el contenido del archivo cargado. ● La estructura final de datos	

					debe permitir ejecutar predicciones y ver los resultados	
HU-02	Detección y Clasificación de Eventos Sospechosos	60	Crítica	Como especialista en redes Wi-Fi, quiero que el sistema detecte y clasifique eventos sospechosos como ataques DoS, para actuar de manera preventiva.	• Investigar y definir patrones y algoritmos para ataques DoS • Implementar el motor de detección de ataques DoS. • Desarrollar la lógica de clasificación de eventos sospechosos.	HU-01

HU-03	Almacenamiento de Ataques Detectados	30	Alta	Como administrador del sistema, necesito que los ataques detectados sean almacenados y acumulados para análisis forense posterior.	• Diseñar el esquema para registrar ataques. • Implementar la capa de persistencia y almacenamiento. • Configurar la funcionabilidad de acumulación de datos.	HU-02
HU-04	Visualización de Estadística de Tráfico	50	Alta	Como usuario del sistema, quiero visualizar gráficos estadísticos (pastel y matriz de confusión) para entender fácilmente el	• El sistema debe registrar cada ataque detectado. • Los registros deben guardarse de forma consultable y persistente.	HU-01, HU-02

				estado del tráfico analizado.	• La información almacenada debe mantenerse tras reiniciar el sistema.	
HU-05	Análisis Específico para Wi-Fi	70	Crítica	Como investigador en seguridad de redes inalámbricas, quiero que el sistema se base en características compatibles con IEEE 802.11 para asegurar que el análisis sea específico al entorno Wi-Fi.	• Investigar y seleccionar características IEE 802.11 relevantes para el análisis • Adaptar el motor de análisis para procesar datos. • Implementar la lógica para distinguir tráfico legítimo de	HU-01

					anómalo en entornos Wi-Fi.	

Requisitos funcionales

En base a los objetivos del sistema y al contexto de redes Wi-Fi bajo el estándar IEE 802.11, se implementaron las siguientes funcionalidades.

Tabla 4

Requisitos Funcionales

Código	Requisito Funcional
RF-01	El sistema debe permitir la **carga de archivos CSV** exportados desde capturas de tráfico Wi-Fi mediante Wireshark.
RF-02	El sistema debe **clasificar automáticamente** el tráfico como "Normal" o "DoS", utilizando un modelo de aprendizaje automático previamente entrenado.
RF-03	El sistema debe generar una **tabla detallada** con las predicciones por flujo de tráfico.
RF-04	El sistema debe mostrar una **gráfica de pastel** con la distribución de tráfico normal y malicioso.
RF-05	El sistema debe permitir al administrador de red visualizar el rendimiento del modelo de detección mediante una matriz de confusión
RF-06	El sistema debe **almacenar de forma acumulativa los registros clasificados como DoS** en un archivo histórico para posteriores auditorías.
RF-07	El sistema debe ofrecer una interfaz web amigable que permita operar el sistema desde navegadores comunes como Chrome.

Requisitos no funcionales

Durante el desarrollo, XP permitió abordar simultáneamente aspectos críticos no funcionales que garantizaron la calidad del sistema.

Tabla 5

Requisitos No Funcionales

Código	Requisito No Funcional
RNF-01	El sistema debe procesar archivos de hasta 50 **MG** en un tiempo máximo de 5 minutos en un entorno con especificaciones como: Intel Core i5 o más, 8 de RAM y almacenamiento SSD.
RNF-02	El sistema debe ser modular, permitiendo actualizar el modelo de machine learning sin modificar la lógica general del backend. Corresponde al atributo de calidad de modificabilidad, para soportar cambios de manera eficiente.
RNF-03	La aplicación web debe ser **responsiva y compatible con múltiples navegadores**, asegurando la usabilidad multiplataforma.
RNF-04	El sistema debe validar los archivos cargados, permitiendo únicamente archivos CSV estructurados correctamente para evitar errores o vulnerabilidades.
RNF-05	El código del sistema debe seguir buenas prácticas de claridad, legibilidad y separación de responsabilidades.

RNF-06	La interfaz se debe diseñarse siguiendo principios de simplicidad y accesibilidad, evitando la sobrecarga visual innecesaria para garantizar una experiencia de usuario eficiente.
RNF-07	El sistema debe garantizar la persistencia y conservación segura de los datos relativos a eventos maliciosos detectados, asegurando que dicha información permanezca íntegra y accesible incluso tras reiniciar la aplicación.

Iteración Inicial – PLANIFICACIÓN (XP)

Durante la iteración inicial, se llevaron a cabo varias actividades para dar el inicio con el proyecto. Se identificaron ideas fundamentales y se definieron los objetivos principales del proyecto. Esta etapa permitió establecer una visión clara del alcance y los resultados.

Se realizó una planificación detallada para guiar el desarrollo posterior del sistema. Para este propósito, se utilizó la Tabla 3, que presenta las tareas derivadas de cada historia de usuario.

Los objetivos principales fueron:

- Identificar las ideas clave.
- Planificar las tareas.
- Establecer los cimientos.
- Definir el flujo de trabajo.

Se estableció una planificación detallada de las tareas del proyecto, basada en las historias de usuario definidas. El entorno de desarrollo fue configurado previamente con las herramientas

Tabla 6

Análisis de las Historias de Usuario

PRIORIDAD	**NÚMERO DE HISTORIAS DE USUARIO**	**ID DE TAREAS**	**ACTIVIDADES**	**HORAS ESFUERZO**
Crítica	**HU-02**	T2-1	Investigar y definir patrones y algoritmos	**10**

			para ataques DoS	
		T2-2	Implementar el motor de detección de ataques DoS	**15**
		T2-3	Desarrollar la lógica de clasificación de eventos sospechosos	**10**
Crítica	HU-05	T5-1	Investigar y seleccionar características IEE 8 relevantes para el análisis	**15**
		T5-2	Adaptar el motor de análisis para procesar datos	**10**
		T5-3	Implementar la lógica para distinguir tráfico legítimo de anómalo en entornos Wi-Fi	**10**
Alta	HU-01	T1-1	Diseñar y desarrollar la interfaz de carga de archivos CSV.	**10**

		T1-2	Implementar la lógica de lectura y procesamiento de CSV	**15**
		T1-3	Configurar la estructura de datos para la clasificación	**10**
Alta	HU-04	T4-1	Diseñar la interfaz de usuario para la visualización de estadísticas	**10**
		T4-2	Desarrollar el componente de generación de gráficos de pastel	**10**
		T4-3	Desarrollar el componente de generación de matriz de confusión	**15**
Alta	HU-03	T3-1	Diseñar el esquema para registrar ataques	**10**
		T3-2	Implementar la capa persistencia y almacenamiento	**10**
		T3-3	Mantener de forma segura todos los datos	**5**

			de ataques registrados	

Una vez que se definió el producto backlog con la duración y prioridad de las historias de usuario, se procedió a distribuirlas en 5 Sprint. Cada iteración tuvo una duración de 2 semanas, trabajando 5 dias a la semana 5 horas.

La Historia de Usuario HU-02, al ser la de prioridad Crítica, fueron consideradas factores centrales del proceso. La distribución de las tareas y las historias de Usuario se detalla en la tabla 7 para así cumplir con la duración y planificación de las iteraciones.

Tabla 7

Distribución de Iteraciones

Iteración	**TAREAS**	**ESFUERZO (HORAS)**	**HISTORIAS DE USUARIO**
1	T2.1, T2.2, T1.1	35	HU-02, HU-01
2	T2.3, T5.1, T5.2	35	HU-02, HU-05
3	T5.3, T1.2, T1.3	35	HU-05, HU-01
4	T4.1, T4.2, T4.3	35	HU-04
5	T3.1, T3.2, T3.3	25	HU-03

Después de finalizar cada iteración, se implementaron las mejores prácticas con el fin de garantizar cada funcionalidad cumpliera con los objetivos. Esto facilitó la evolución del sistema, asegurando su alineación con las necesidades del proyecto

Arquitectura del sistema

La arquitectura del sistema se basó en un modelo cliente-servidor simple, en el cual el usuario interactúa a través de una interfaz web, mientras que el servidor se encarga del procesamiento de datos y la generación de predicciones mediante el modelo de aprendizaje automático. A continuación, se presenta la descripción general de la arquitectura del sistema.

Esta arquitectura del sistema, fundamentada en el modelo cliente-servidor, permitió una separación clara entre tres componentes principales: la interfaz de usuario, el procesamiento de datos y el motor de inferencia basado en aprendizaje automático. Esta

división favorece la escalabilidad del sistema, ya que esto posibilita ampliar o mejorar la capacidad del procesamiento del servidor sin afectar la interfaz de usuario. Asimismo, incrementa la modificabilidad, dado que cada componente puede actualizarse o reemplazarse de manera independiente, sin comprometer el funcionamiento del sistema.

A continuación, en la Tabla 8 se muestran los elementos de la arquitectura del sistema

Tabla 8

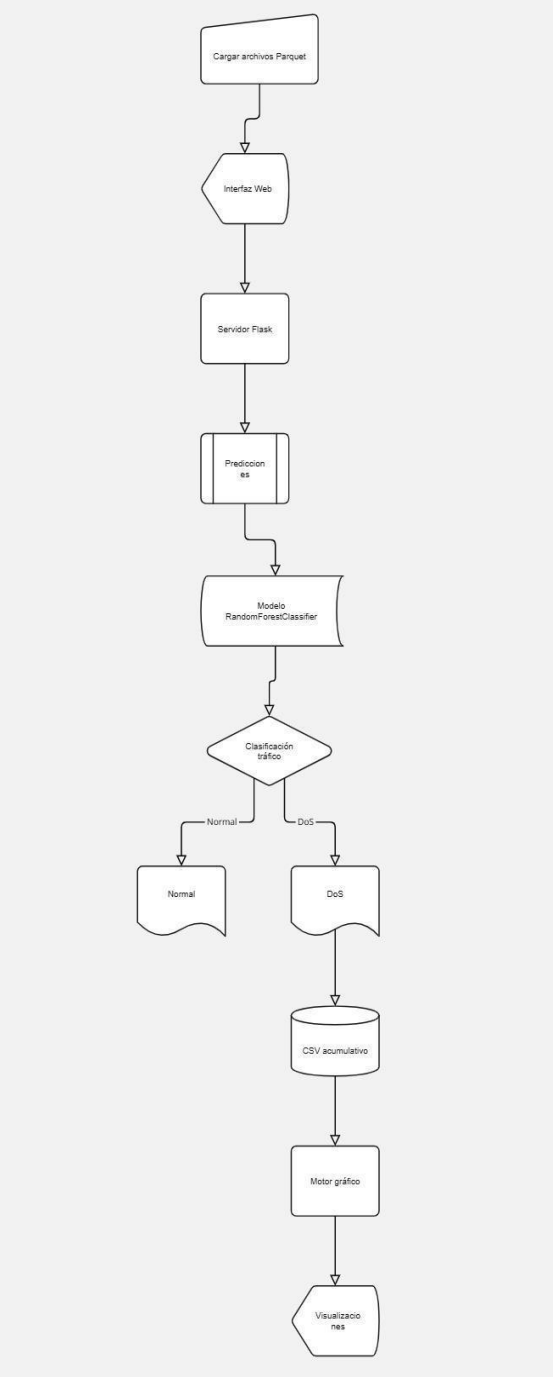
Arquitectura del Sistema

Arquitectura del Sistema	Descripción
Frontend (Interfaz Web)	Interfaz desarrollada en HTML y CSS utilizando Bootstrap. Permite al usuario cargar archivos Parquet y visualizar los resultados (tablas, gráficas).
Backend (Servidor Flask)	Aplicación desarrollada en Python con Flask. Se encarga de recibir los archivos cargados, realizar las predicciones con el modelo entrenado, generar gráficos, y devolver resultados al usuario.
Modelo de Machine Learning (modelo_dos.pkl)	Modelo previamente entrenado con un algoritmo RandomForestClassifier que clasifica el tráfico de red en "Normal" o "DoS", a partir de características específicas extraídas del estándar IEEE 802.11.
Base de datos temporal (CSV acumulativo)	Archivo CSV ubicado en el servidor, que almacena de forma acumulativa los registros identificados como ataques DoS para auditoría o análisis forense.
Motor gráfico	Módulo basado en Matplotlib para generar visualizaciones (gráfico de pastel y matriz de confusión) dinámicamente, y presentarlas en la interfaz web.

En la ilustración 5 se muestra una representación en un diagrama de flujo

Ilustración 5

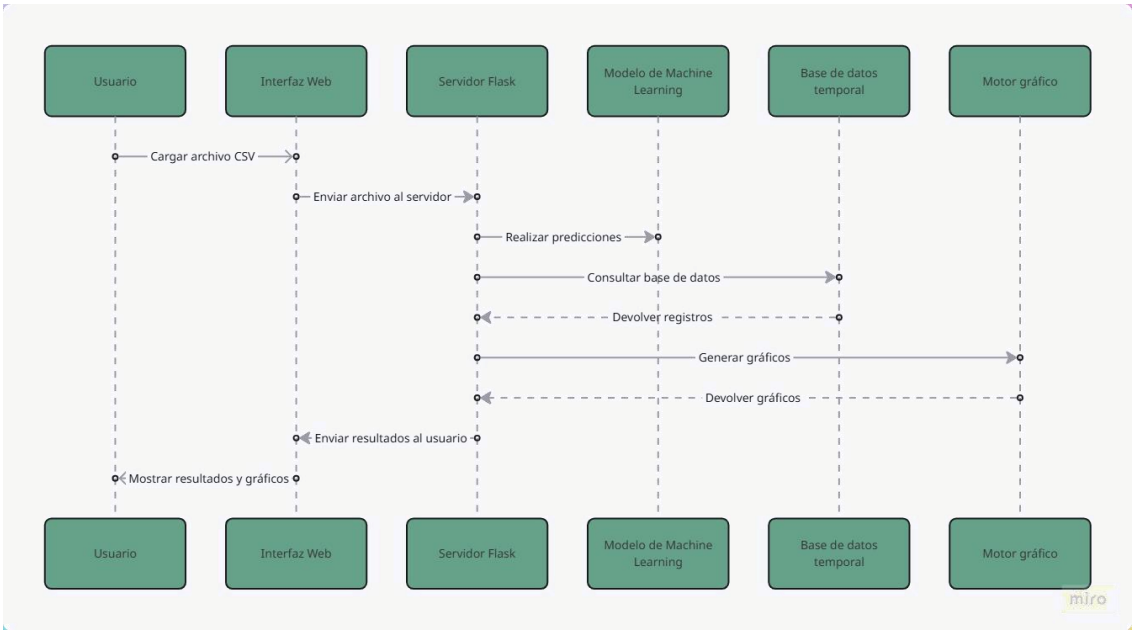
Diagrama de flujo



En la ilustración 6 se muestra el diagrama de secuencia que representa el flujo de interacción entre los componentes del sistema durante el proceso de análisis y clasificación del tráfico.

Ilustración 6

Diagrama de secuencia



Diseño Visual y Lógico

La interfaz de usuario se desarrolló utilizando HTML y CSS, complementados con Bootstrap para asegurar un diseño limpio, moderno y responsivo. El backend fue implementado en Python empleando el framework Flask, encargado de gestionar la lógica de procesamiento: recibir los archivos CSV, cargar el modelo previamente entrenado, realizar las predicciones y enviar los resultados a la interfaz para su visualización.

2.4 Codificación – Configuración

Configuración del entorno

Se implementó una red inalámbrica controlada utilizando un punto de acceso UniFi gestionado por UniFi Controller. Esto permitió hacer varias pruebas de conectividad bajo condiciones normales y en presencia de tráfico malicioso simulado.

En la Ilustración 7 se muestra el entorno de trabajo donde se llevó a cabo la configuración de la red, mientras que en la Tabla 9 los componentes físicos utilizados.

Tabla 9

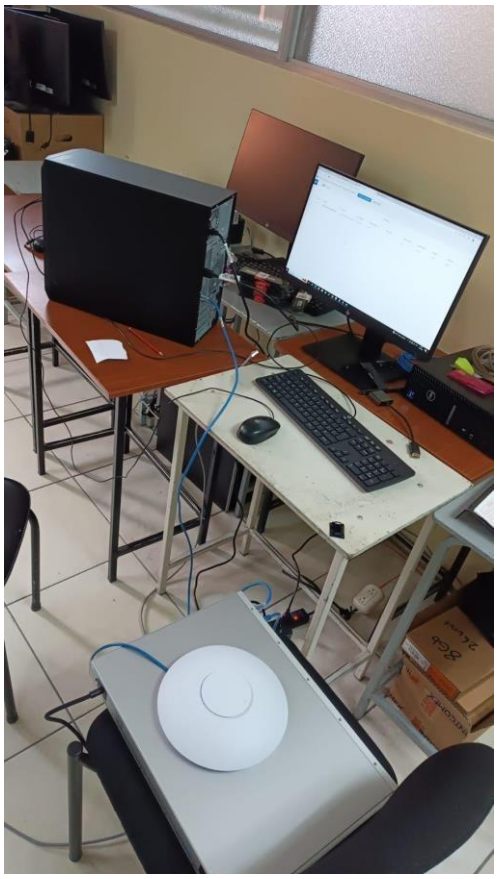
Componentes

Componente	Descripción
PC	Estación de trabajo con 8 GB de RAM y 512 de almacenamiento, utilizado para ejecutar

	las herramientas de monitoreo, generación de tráfico y el servidor del sistema.
Sistema Operativo	Windows, ambiente en el que se configuraron las herramientas y se desarrollaron los scripts.
Switch	UniFi Enterprise Switch, encargado de la conectividad de red, permitiendo la gestión de VLANs y el control de tráfico
Access Point	UniFi UAP-AC-LR, configurado para emitir una red Wi-Fi de prueba, sirviendo como punto de acceso para los dispositivos conectados y objetivo del ataque DoS simulados.

Ilustración 7

Configuración del entorno de pruebas



2.5 Métricas y Metodologías de Evaluación

2.5.1 Evaluación del Algoritmo

Para asegurar la fiabilidad y precisión del algoritmo, su evaluación se realizó mediando el uso de un Dataset que se seleccionó cuidadosamente y la aplicación de métricas para tener un rendimiento estándar en el ámbito del aprendizaje para la clasificación.

El Dataset que fue seleccionado fue CICDDoS2019 por sus características relevantes: tamaño, variedad de ataques, representación de tráfico de red. La cual contiene una gran colección de tráfico de red, las características incluidas en el Dataset: duración, bytes por segundo, número de paquetes, Flags TCP/UPD entre otros. Lo que permitieron entrenar y probar el algoritmo con una representación del comportamiento de la red.

El Dataset fue dividido en conjuntos de entrenamiento, validación y prueba para asegurar una evaluación imparcial del rendimiento.

Se implementó la validación cruzada debido a que los sistemas de detección de ataques DoS requieren una validación rigurosa. La validación cruzada permite utilizar todos los datos disponibles para evaluar el modelo, proporcionando una estimación más precisa por una única partición aleatoria. Además, al usar k=5 se asegura que cada tipo de ataques DoS esté representado proporcionalmente en cada iteración de entrenamiento y evaluación. Esto facilitó una mejor estabilidad en el modelo y garantizó que su desempeño sea consistente en un entorno real de red Wi-Fi.

La evaluación de los algoritmos se centró en las siguientes medidas de evaluación, tales como:

- **Matriz de Confusión:** Es una tabla que resume el rendimiento de un algoritmo de clasificación en un conjunto de datos de prueba. Donde:
    - **Verdaderos Negativos (TN- True Negatives):** Son los casos en los que el algoritmo predijo correctamente que el tráfico correspondía a un ataque DoS y, en efecto, era un ataque DoS.
    - **Verdaderos Negativos (TN- True Negatives):** Son los casos en los que el algoritmo predijo correctamente que el tráfico era normal y, en efecto, era tráfico normal.

- **Falsos Positivos (FP-False Positives):** Son los casos en los que el algoritmo predijo incorrectamente que el tráfico era un ataque DoS, cuando en realidad era tráfico normal.
- **Falsos Negativos (FN-False Negatives):** Son los casos en los que el algoritmo predijo incorrectamente que el tráfico era normal, cuando en realidad era un ataque DoS.

- **Precisión (Accuary):** Mide la proporción de predicciones correctas sobre el total de casos. Es útil para una visión general, pero debe completarse con otras métricas en Dataset desbalanceados.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precisión:** Indica la proporción de instancias positivas que fueron correctamente identificadas como positivas. Es crucial para minimizar los falsos positivos, que en un sistema de detección temprana reducirían la confianza del administrador.

$$Precision = \frac{TP}{TP + FP}$$

- **Sensibilidad / Recall:** Mide la proporción de instancias positivas reales que fueron correctamente identificadas como positivas. En la detección de ataque DOS.

$$Recall = \frac{TP}{TP + FN}$$

- **Puntuación F1 (F1-Score):** Es la media armónica de la precisión y la sensibilidad, proporcionando una medida equilibrada del rendimiento del modelo, especialmente útil cuando las clases están desbalanceadas.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

- **Curva ROC (Receiver Operating Characteristic) y AUC (Area Under the Curve):** La curva ROC ilustra el compromiso entre la tasa de verdaderos positivos y la tasa de falsos positivos en diferentes umbrales. El AUC cuantifica el área bajo esta curva, con un valor cercano a 1 indicando un excelente rendimiento del clasificador

2.5.2 Evaluación del sistema

La evaluación del sistema de detección temprana fue fundamental para validar su funcionabilidad, capacidad de respuesta y estabilidad en entornos de red Wi-Fi, asegurando su eficacia ante posibles ataques DoS.

- **Funcionabilidad:** Se verificó que todos los componentes del sistema operaran correctamente y que la plataforma fuera intuitiva y fácil de usar para los administradores.
- **Rendimiento:** Se midió la latencia de detección y la capacidad de procesamiento del sistema bajo diferentes volúmenes de tráfico. Además, se evaluó el impacto en el rendimiento de la red Wi-Fi para asegurar que no introdujera latencia o consumo excesivo de recursos.
- **Resistencia a Ataques DoS:** Lo más importante fue verificar la capacidad del sistema para detectar estos ataques de manera eficaz y mantener su estabilidad operativa bajo cargas extremas.

Pruebas de conectividad y trafico

En esta fase se verificó el funcionamiento de la red, comprobando la conectividad entre dispositivos conectados a través del punto de acceso. Posteriormente, se procedió a generar tráfico de red controlado, incluyendo tanto tráfico legitimo como tráfico malicioso, utilizando herramientas específicas para emular condiciones reales de uso y ataque.

Monitoreo y captura de tráfico

Durante esta etapa, se procedió a la captura de tráfico de red utilizando la herramienta Wireshark, previamente instalada en la estación de trabajo. Esto permitió observar los

paquetes transmitidos y registrar información relevante para su análisis posterior, tales como las direcciones IP de origen y destino, el protocolo utilizado, la cantidad de bytes transferidos y el tiempo de duración de cada transmisión.

2.6 Tecnologías contempladas

El diseño conceptual se basó en un conjunto de herramientas especializadas en el análisis de tráfico de red, aplicando simulaciones tanto en entornos controlados como en la implementación de modelos mediante técnicas de aprendizaje automático.

A continuación, se presentan las tecnologías sugeridas:

- **Wireshark:** Herramienta usada para capturar el tráfico de la red en formato. pcap el cual fue exportado y convertido a .csv para ser utilizado como entrada del sistema. (Foundation T. W., 2024)
- **Python:** Es un lenguaje de programación orientado a objetos de alto nivel y fácil de interpretar con una sintaxis fácil de leer. (Foundation P. S., Python, 2024)
- **Dask:** Es una librería diseñada para el procesamiento de grandes volúmenes de datos en formato Parquet que permite analizar de manera distribuida y eficiente. (Team D. D., 2023)
- **Pandas y Numpy:** Librerías fundamentales para el manejo, transformación y análisis de datos estructurados, utilizadas en la preparación de características del Dataset (team, 2023).
- **Os:** Este módulo es una biblioteca estándar de Python que permite interactuar con el sistema operativo para gestionar archivos, directorios y variables de entorno. (Foundation P. S., Python Docs, 2024)
- **Pathlib:** Es una biblioteca moderna de Python para manejar las rutas de archivo de forma orientada a objetos. (Foundation P. S., Python, 2024)
- **Shutil:** Módulo estándar de Python utilizado para realizar operaciones de alto nivel sobre archivos y colecciones de archivos, como copiar o eliminar directorios completos. (Team C. , 2023)
- **Sys:** Módulo estándar de Python que nos proporciona acceso a parámetros y funciones específicas del sistema, útil para la interacción con el intérprete y el entorno de ejecución. (Foundation P. S., Python Documentation, 2024)

- **Logging:** Módulo estándar de Python que permite registrar mensajes de estado, depuración, y error, facilitando el seguimiento y la depuración del flujo de ejecución del sistema. (Foundation P. S., Python Documentation, 2024)
- **PyArrow (Parquet):** Biblioteca fundamental que proporciona herramientas para leer y escribir eficientemente archivos en formato Parquet, optimizada para grandes volúmenes de datos y el intérprete Dask y Pandas para un procesamiento de datos. (Foundation A. S., 2024)
- **Scikit-learn:** Biblioteca empleada ampliamente en la creación de los modelos de clasificación y para la detección de anomalías. (Pedregosa, 2023)
- **Random Forest Classifier:** Se trata de un algoritmo de machine learning fundamentado en árboles de decisión, elegido por su solidez y alto rendimiento en tareas de clasificación binaria, especialmente en escenarios de detección de tráfico anómalo como son los ataques de denegación de servicio. (Pedregosa, 2023)
- **LinearSVC:** Clasificador basado en Máquinas de Vectores de Soporte (SVM), utilizado por su eficacia en la clasificación y su capacidad para manejar datos grandes. (Apache Software Foundation, 2024)
- **XGBoost (XGBoostClassifier):** Implementación optimizada de algoritmos de Gradient Boosting, reconocida por su alta velocidad y rendimiento en una amplia gama de tareas de clasificación. (IBM Developer Team, 2023)
- **LightGBMClassifier:** Algoritmo de Gradient Boosting basado en árboles, seleccionado por su notable eficiencia y velocidad en el entrenamiento con grandes volúmenes de datos. (Team), 2025)
- **Imblearn (RandomUnderSampler** (imbalanced-learn Developers, 2024)**):** Es una biblioteca que proporciona un balanceo de clases en conjuntos desequilibrados, es esencial para mitigar el sesgo del modelo.
- **Joblib:** Es una herramienta que se utiliza para la serialización y deserialización eficiente de objetos Python, permitiendo guardar y cargar el modelo entrenado el escalador y lo que es el codificador sin necesidad de reentrenar el sistema en cada ejecución. (Developers, 2023)
- **Flask:** Micro-framework web ligero en Python, utilizando para construir el backend, facilitando la creación de rutas, el manejo de peticiones HTTP y la interacción con el modelo de aprendizaje automático. (Grinberg, 2022)

- **Flask-CORS:** Es una extensión de Flask que implementa Cross-Origin Resource Sharing (CORS), hace que lo recursos del backend sean solicitados, para la interacción segura. (Flask-CORS Developers, 2024)
- **Matplotlib:** Biblioteca de Python para la creación de visualizaciones estáticas, animadas e interactivas. Fueron utilizados para la generación de los graficos para presentar en la interfaz web. (Hunter, matplotlib.org, 2023).
- **Seaborn:** Es una biblioteca de visualización de datos basada en Matplotlib, que proporciona una interfaz de alto nivel para gráficos estadísticos. (Seaborn Developers, 2024)
- **Bootstrap:** Framework CSS y JavaScript utilizado para el desarrollo del Frontend del sistema web, proporcionando un diseño responsivo y predefinido que acelera la creación de interfaces de usuarios modernas y accesibles. (Authors, 2023)

CAPITULO III

Resultados y Discusión

Finalizado el desarrollo del proyecto, en este capítulo se presentan de manera detallada los resultados alcanzados en cada una de las fases de la investigación. De esta forma, se demuestra el cumplimiento de los objetivos, validando la metodología aplicada y evidenciando la efectividad de las soluciones.

3.1 Resultados fase 1. Análisis de los algoritmos de predicción

Se evaluaron los cuatro modelos utilizando diversas métricas, se empleó la función *dump* de la librería *joblib*, esta librería permite almacenar cada modelo en un archivo con extensión. *pkl*, especificando tanto el objeto del modelo como el nombre del archivo.

A continuación, se presentan los resultados de dichas evaluaciones:

3.1.1 Modelo Random forest

En las Ilustración 8 se muestra que el modelo Random Forest Classifier alcanzó una exactitud del 81.63%, una precisión ponderada del 95.93%, un recall ponderado del 81.63%, y un F1-Score ponderado del 85.03%. Aunque estas métricas sugieren un buen desempeño, el análisis detallado por clase reveló variaciones en su fiabilidad. El modelo mostró una alta eficacia para identificar clases como ´LDAP´, ´MSSQL´ y ´Syn´, con F1 Scores de 0.95, 0.94 y 1.00 respectivamente. Sin embargo, el rendimiento fue considerable para las categorías como ´Portmap´ (F1-Score de 0.10) y ´UDPLAG´ (F1-Score de 0.01), donde a pesar de un alto recall, la precisión fue muy baja. De igual manera, la clase ´NetBIOS´ presentó un recall de solo 0.32.

En la Ilustración 9 se representa la matriz de confusión generada por el modelo Random Forest Classifier, los valores de la diagonal principal representan valores de verdaderos positivos (TP), es decir, las instancias correctamente clasificadas, destacando un rendimiento sobresaliente en categorías ´LDAP´ (10,852 instancias),´MSSQL´ (31,752 instancias) y ´Syn´ (29,285 instancias), que presentan altos TP, lo que se correlaciona directamente con los altos F1-Scores previamente reportados (0.95, 0.94 y 1.00 respectivamente), subrayando una robusta capacidad de generalización predictiva.

Los valores que estan fuera de la diagonal principal en la matriz de confusión estan divididos en Falsos Positivos (FP), y Falsos negativos, cuando no se identifica

correctamente una instancia de su clase verdadera, un FP se observa si una instancia real ´MSSQL´es clasificada como ´LDAP, mientras que un FN ocurre cuando una instancia de ´NetBIOS´ es erróneamente predicha como ´Portmap´. El análisis revela deficiencias importantes, como en la clase ´Portmap´, que pese a contar con 878 verdaderos positivos, presenta 231 FN hacia ´NetBIOS´, lo que contribuye a un bajo F1-Score de 0.10; en ´UDPLag´, con solo 10 verdaderos positivos y una alta cantidad de FP provenientes de ´MSSQL´y ´UDP´, explicando su íntimo F1-Score de 0.01; y en ´NetBIOS´, que aunque alcanza 7.051 TP, sufre 14.823 FN clasificados como ´Portmap´, reduciendo su recall a 0.32 y evidenciando una limitación significativa en la capacidad discriminativa del modelo para esta clase. En conjunto, la matriz de confusión es crucial no solo para medir la precisión global, si no tambien para identificar patrones específicos de confusión entre clases, orientando así mejoras en el algoritmo y en las características de entrada para fortalecer la robustez del clasificador.

Ilustración 8

Modelo Random Forest Classifier

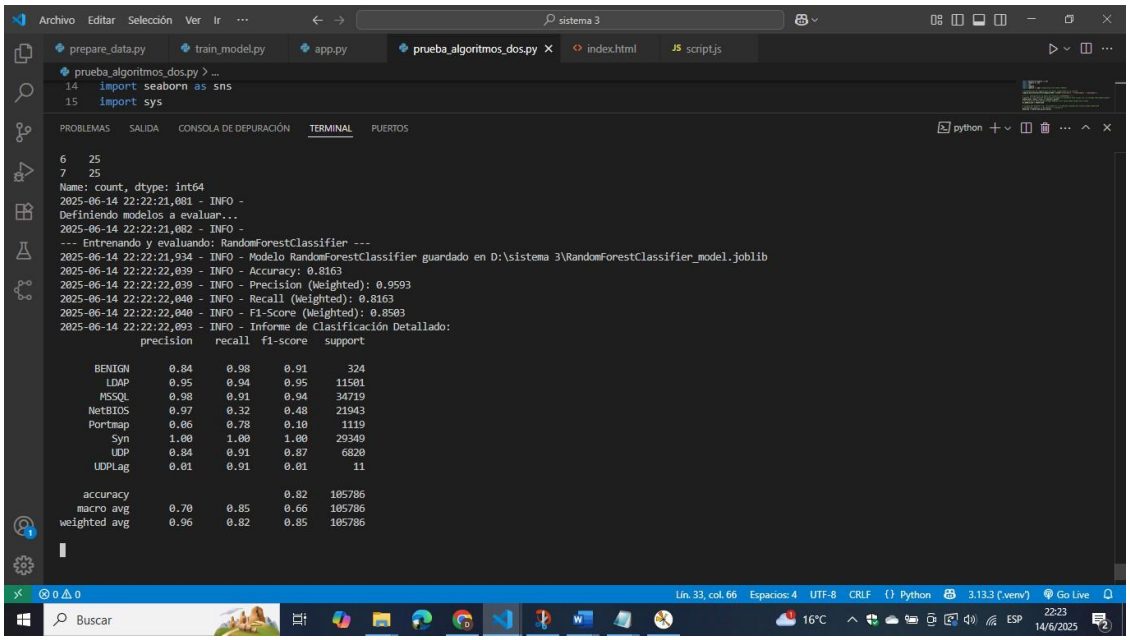
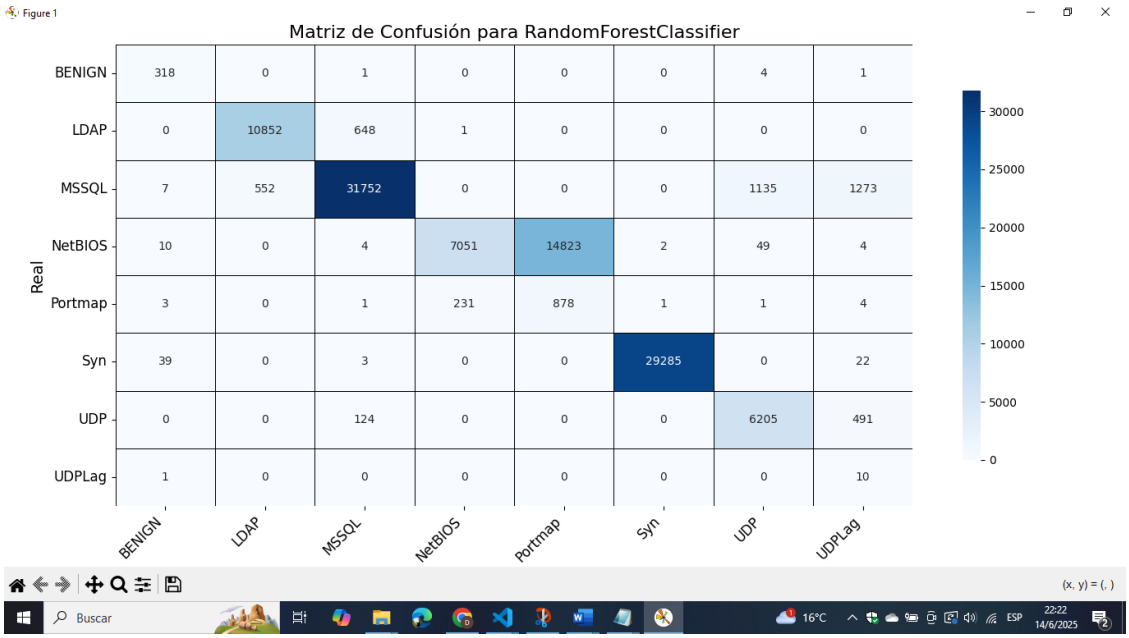


Ilustración 9

Matriz de Confusión Random Forest Classifier



3.1.2 Modelo Linear Support Vector Classifier (SVC)

En las Ilustración 10 se obtuvo una exactitud del 68.02%, con una precisión ponderada del 91.88%, un recall ponderado del 68.02%, y un F1-Score ponderado del 71.56%. Si bien este modelo fue eficaz para clasificar ´**LDAP**´ (F1-Score 0.93) y ´**Syn**´ (F1-Score 0.99), su desempeño fue significativamente deficiente en otras categorías. Específicamente, para '**NetBIOS**', el recall fue extremadamente bajo (0.07), resultando en un F1-Score de 0.13. '**Portmap**' obtuvo un F1-Score de 0.10 debido a su baja precisión (0.05). Para '**UDPLag**', la precisión fue de 0.00 a pesar de un recall de 1.00, resultando en un F1-Score de 0.00. Además, '**UDP**' mostró un recall bajo (0.49).

En la Ilustración 11 se muestra que la matriz de confusión para el modelo LinearSVC, es fundamental para comprender el rendimiento en un entorno multiclase. Los valores en diagonal principal de la matriz indican Verdaderos Positivos (TP), el número de instancias correctamente clasificadas por el modelo para cada una de las categorías. En este caso, se observa un alto conteo de TP para ´LDAP´ (11,415), ´MSSQL´ (25,775) y ´Syn´ (28,767), lo que sugiere una robusta capacidad del modelo para identificar correctamente las clases.

Los datos fuera de la diagonal principal en la matriz de confusión representan los errores de clasificación, que se dividen en dos tipos fundamentales: Falsos Positivos (FP) y Falsos

Negativos (FN). Un FP ocurre cuando el modelo asigna incorrectamente una instancia a una clase distinta a la real, como 71 instancias reales de ´LDAP´ fueron clasificadas erróneamente como ´MSSQL´. Por otro lado, un FN sucede cuando el modelo no logra identificar correctamente una instancia dentro de su clase verdadera, como, por ejemplo, cuando 7,669 instancias reales de ´MSSQL´ fueron clasificadas como ´UDPLag´. El análisis pone en evidencia que el modelo puede mejorar, cuando en la clase ´NetBIOS´, que presenta un alto número de FN, con 14,811 instancias mal clasificadas como ´Portmap´ y 5,043 como ´UDPLag´ tiene 1,170 FN asignados a ´UDPLag´, y ´MSSQL´ muestra 1,240 FN clasificados como ´LDAP´ y 7,669 como ´UDPLag´. Estos patrones de error, revelados por la matriz son muy esenciales para entender las fortalezas y limitaciones del clasificador Linear SVC.

Ilustración 10

Modelo Linear SVC

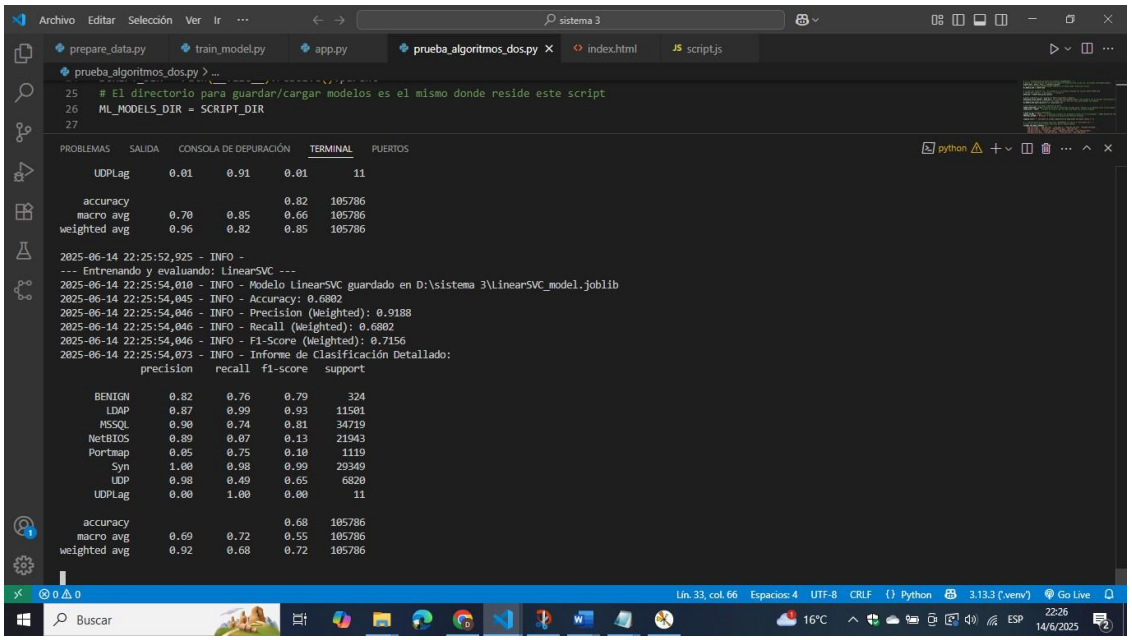
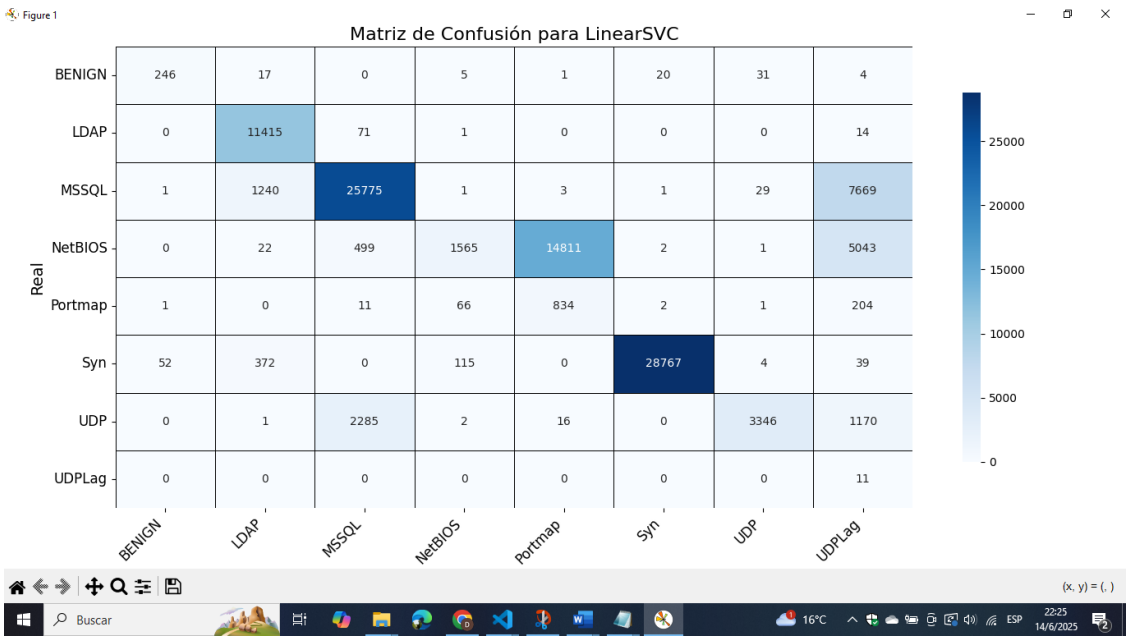


Ilustración 11

Matriz de Confusión Linear SVC



3.1.3 Modelo XGBoostClassifier

En las Ilustración 12 se obtuvo que el modelo XGBoostClassifier nos demostró un rendimiento general con una exactitud del 86.36%, una precisión ponderada del 95.62%, un recall ponderado del 86.36% y un F1-Score ponderado del 89.55%. Este desempeño supera a los demás modelos, reflejando su capacidad para clasificar con alta fiabilidad categorías como ´Syn´ (F1-Score 1.00), ´LDAP´ (F1-Score 0.94) y ´MSSQL´ (F1-Score 0.95), además de la clase ¨BENINGN´ (F1-Score 0.85). Sin embargo, el modelo presentó limitaciones significativas en la identificación de clases minoritarias: ´NETBIOS´ tuvo un recall bajo (0.54) con un F1-Score de (0.69); ´Portmap ´obtuvo un F1-Score de apenas (0.10) debido a su precisión muy baja (0.06) y ´UDPLag ´mostró un F1-Score crítico de (0.01), con una precisión de (0.01) a pesar de un recall de (0.91), indicando una alta tasa de falsos positivos.

En la Ilustración 13 la matriz de confusión para el modelo XGBoostClassifier, se refleja la diagonal principal como Verdaderos Positivos (TP). Para este caso, se destacan valores elevados de TP para las categorías ´LDAP´ (10,839), ´MSSQL´ (32,407) y ´Syn´ (29,283), lo que da como evidencia una capacidad predictiva robusta para estas clases. Esta eficacia se pudo confirmar además con las métricas de precisión y recall altas alcanzando 0.93 y 0.94 para ´LDAP´, 0.98 y 0.94 para ´MSSQL´ y valores perfectos de 1.00 en ambos indicadores para ´Syn´, lo que subraya la efectividad para las categorías.

Los valores fuera de la diagonal principal en la matriz de confusión corresponden a errores de clasificación, lo que dividen en dos tipos; Falsos Positivos (FP) y Falsos Negativos (FN). Un FP ocurre cuando el modelo asigna incorrectamente una instancia a una clase distinta de la real, como sucede cuando 660 instancias reales de ´LDAP´ son clasificadas erróneamente como ´MSSQL´. Por otra parte, un FN se da cuando el modelo no logra identificar una instancia dentro de la clase verdadera, por ejemplo, cuando 858 instancias reales de ´MSSQL´ se predice como ´UDPLag´. El análisis de errores evidencia áreas donde el modelo puede mejorar, especialmente en la clase ´NetBIOS´, que a pesar de contar con 11,811 verdaderos positivos, presenta un elevado número de FN: 9,849 instancias reales se clasificaron como ´Portmap´ y 202 como ´UDPLag´, lo que reduce su recall a 0.51, la clase ´Portmap´ tiene 574 FN asignados a ´NetBIOS´, mientras que ´UDP´ muestra 320 FN clasificados como ´UDPLag´. Estos patrones resaltan la necesidad de ajustar el modelo para mejorar su capacidad minoritaria entre clases con características similares.

Ilustración 12

Modelo XGBoostClassifier

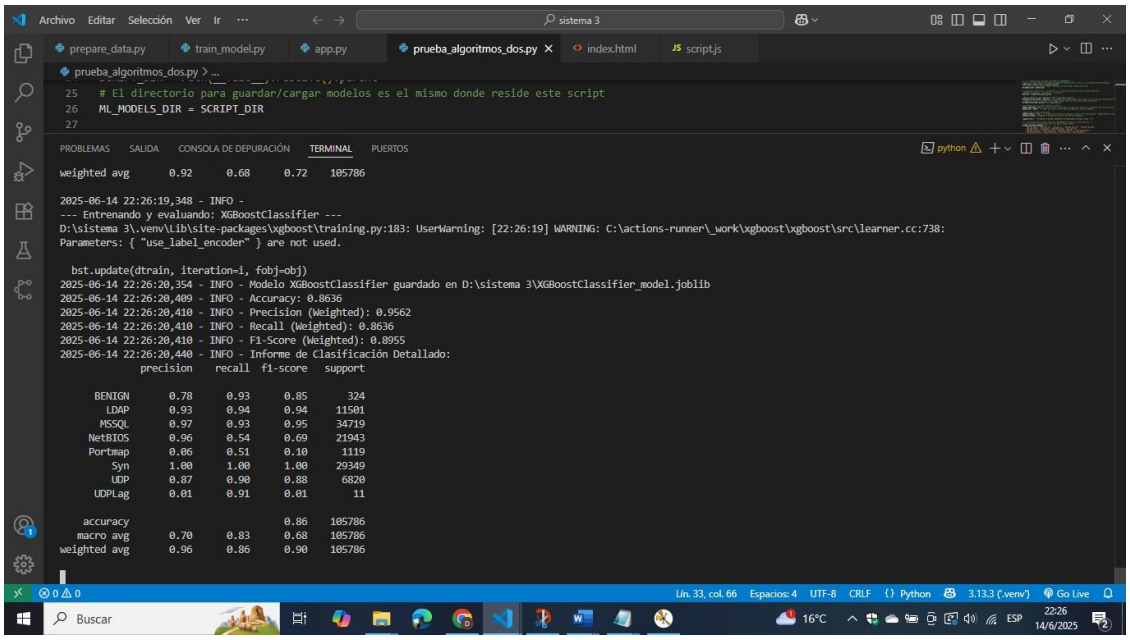
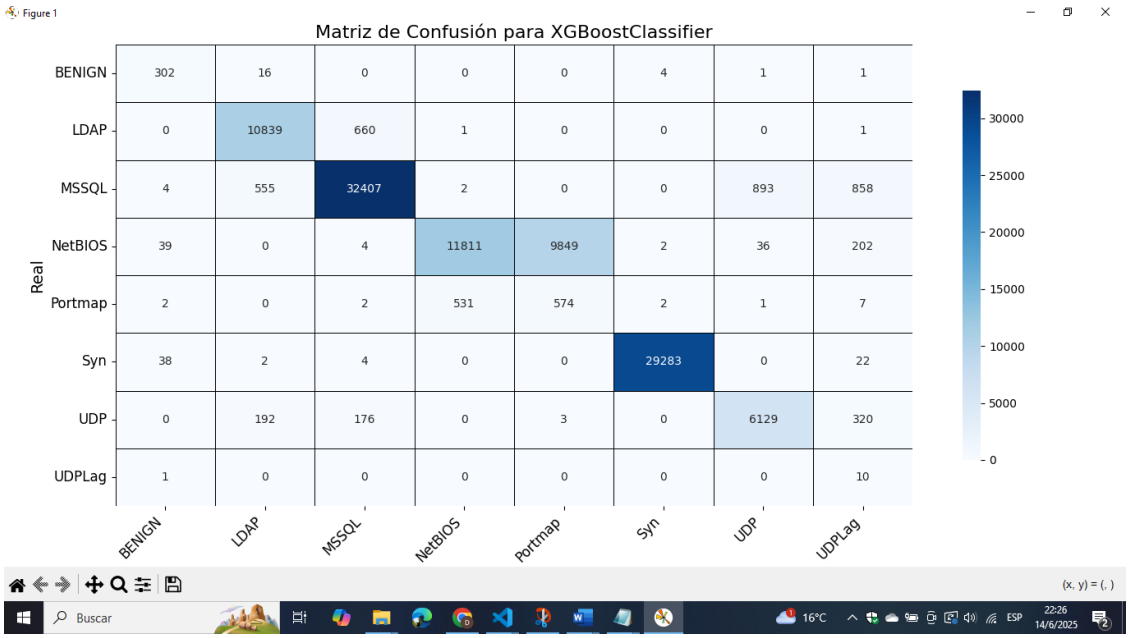


Ilustración 13

Matriz de Confusión para XGBoostClassifier



3.1.4 Modelo LightGBMClassifier

En las Ilustraciones 14 y 15 se obtuvo como resultado que el modelo LightGBMClassifier demostró un sólido desempeño general con una exactitud del 84.72%. Sus métricas ponderadas globales también fueron elevadas con una precisión ponderada del 95.70%, un recall ponderado del 84.72%, y un F1-Score ponderado del 88.24%. Lo que estas cifras posicionan a LightGMB como un clasificador altamente competitivo, comparable con los resultados obtenidos por XGBoost y Random Forest.

Ilustración 14

Modelo LightGBMClassifier

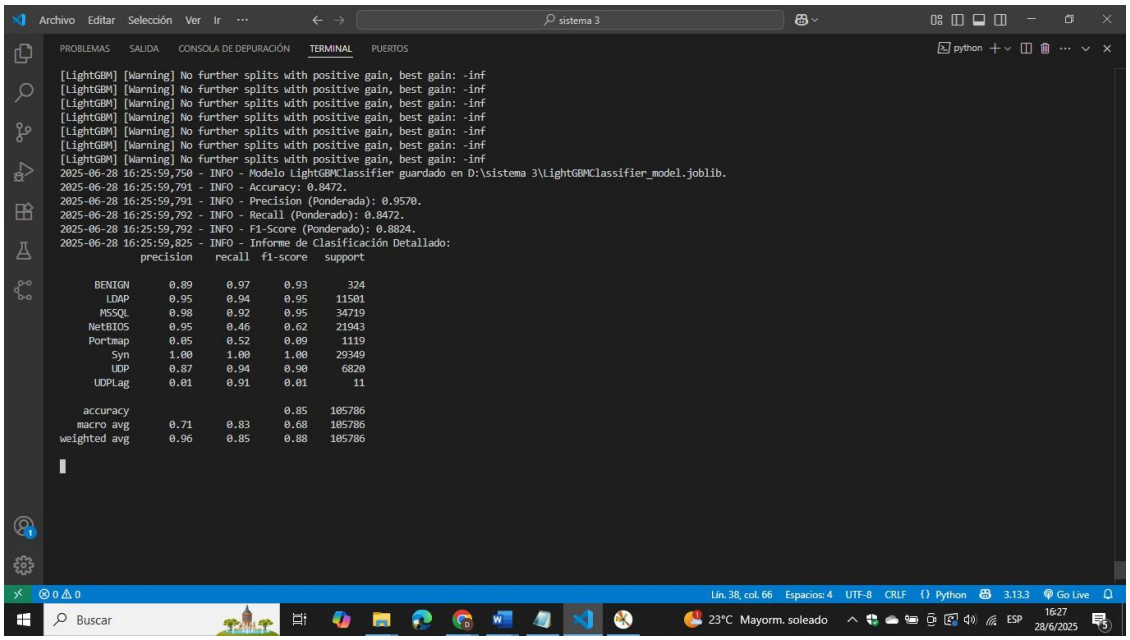
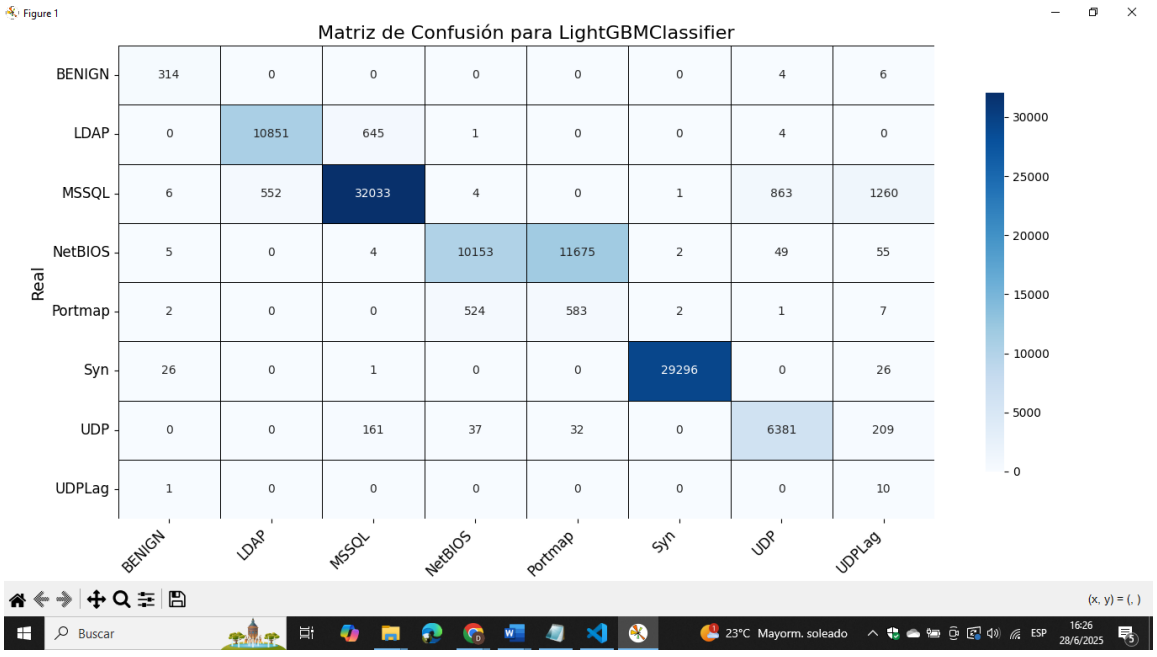


Ilustración 15

Matriz de Confusión LightGBMClassifier



3.1.5 Selección del modelo

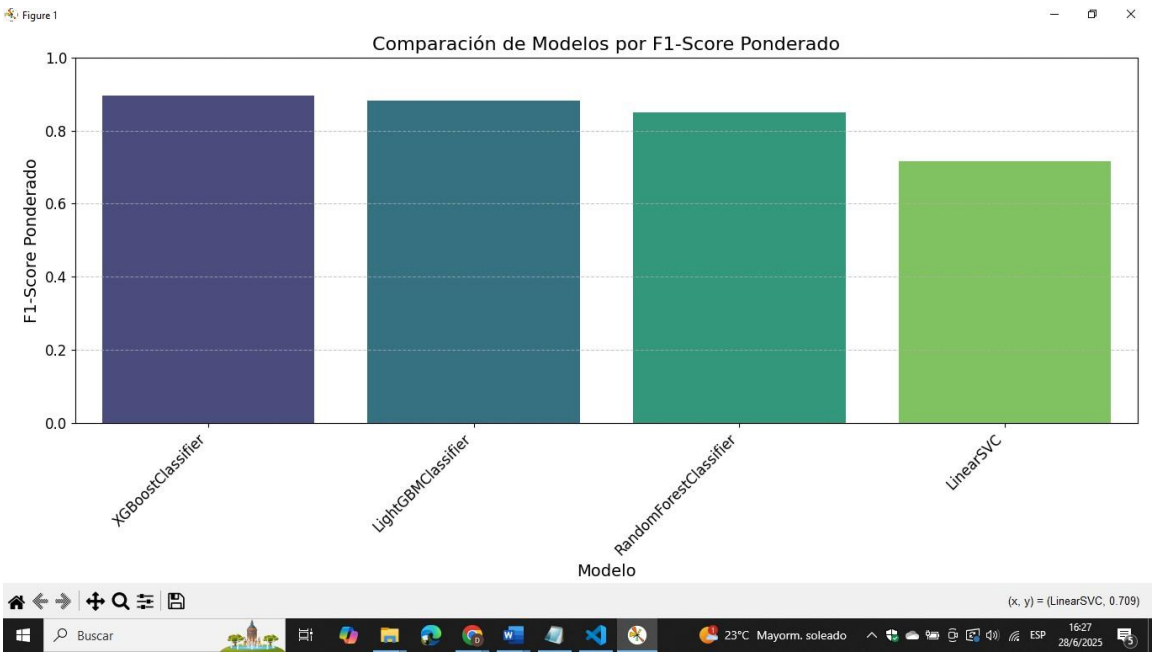
La evaluación de los cuatro modelos de clasificación evidencio muchas diferencias en el desempeño para abordar el problema planteado. El modelo XGBoostClassifier obtuvo el mejor rendimiento global, alcanzando una exactitud del 86.36% y un F1-Score ponderado del 89.55%, lo que lo posiciona como el modelo más efectivo en el conjunto de datos de

prueba. Luego muy cercano está el modelo LightGMBClassifier mostró resultados sobresalientes con una exactitud del 84.72% y un F1-Score ponderado del 88.24%, consolidándose como una alternativa competitiva. Por otra parte, el modelo RandomForestClassifier también presentó un rendimiento robusto, con una exactitud del 81.63% y un F1-Score ponderado del 85.03%. El modelo LinearSVC registró el desempeño más bajo entre los clasificadores evaluados, con una exactitud del 68.02% y un F1-Score ponderado del 71.56%, lo que sugiere que su enfoque lineal podría no ser el más adecuado para la complejidad y características del conjunto de datos utilizado.

A continuación, en la Ilustración 17 se muestra el resumen:

Ilustración 16

Análisis comparativo de los Modelos



La selección del modelo ***RandomForestClassifier*** para el desarrollo del sistema, a pesar de que algoritmos Boosting como XGBoost y LightGBM mostraron métricas marginalmente superiores en la fase preliminar, se fundamentó en una evaluación integral que se consideró no solo la precisión predictiva, sino también la robustez, capacidad de generalización e interpretabilidad del modelo en un contexto aplicado.

Si bien XGBoost y LightGbm son algoritmos de Gradient Boosting que construye modelos de forma secuencial, donde cada nuevo árbol puede corregir el error anterior. Esta es una de las estrategias que hace alcanzar una alta precisión y eficiencia en la predicción. Debido a la naturaleza secuencial y a que cada árbol depende del anterior,

estos modelos son más propensos al sobreajuste cuando no llegan a ajustarse a los hiperparámetros. Por otro lado, Random Forest está basado en bagging (boostrap aggregating) y en aleatoriedad en la selección de características para construir múltiples árboles independientes y luego combinar los resultados. La independencia entre árboles reduce la varianza del modelo lo que confiere a un mayor sobreajuste, dando como resultado que Random Forest tenga más estabilidad frente a los cambios en los conjuntos de los datos.

Esta característica resulta crucial para garantizar que el modelo mantenga un desempeño consistente y confiable en escenarios reales, más allá de su rendimiento en datos de prueba. Además, la arquitectura de Random Forest facilita una mejor interpretabilidad en comparación con los modelos de Boosting.

Finalmente, la capacidad de procesamiento paralelo de Random Forest representa una ventaja computacional significativa en conjuntos de datos grandes. La elección de Random Forest se alinea con una estrategia de modelado que prioriza la confiabilidad a lo largo plazo y la capacidad de generalización, asegurando que el modelo sea mucho más robusto y menos propenso a problemas en su implementación práctica. (Berríos, Garcia, Hermosilla, & Allende-Cid, 2025)

3.2 Desarrollo de Entrenamiento del Modelo

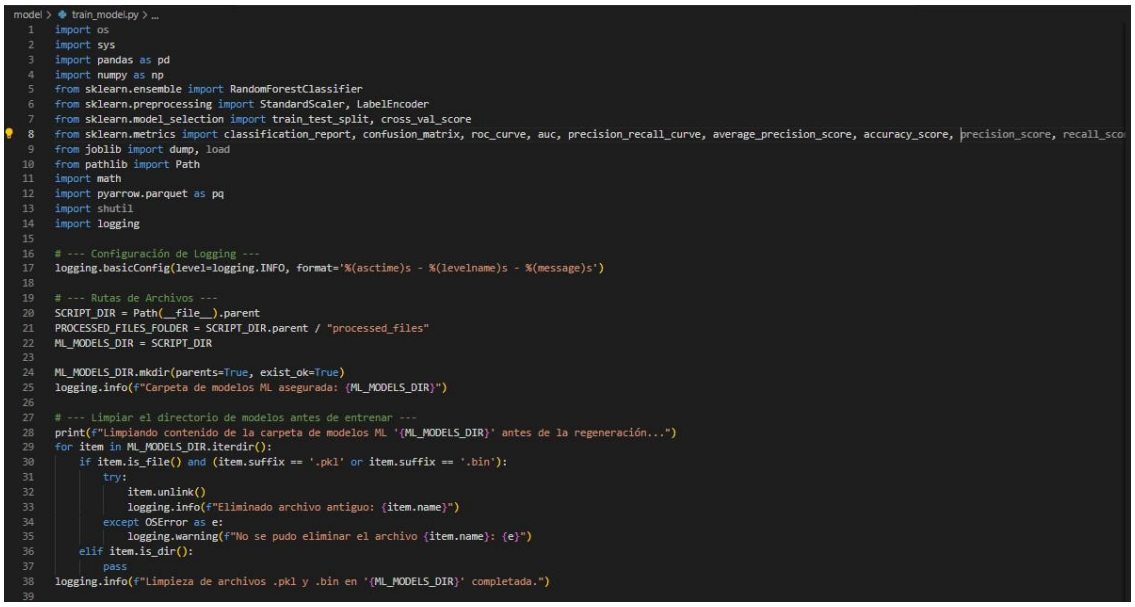
Se detalla la implementación de entrenamiento del modelo de Machine Learning, el cual es el componente central del sistema para la detección de ataques. Se describen los pasos clave del proceso, incluyendo la configuración del entorno, la preparación (escalado y codificación), el entrenamiento del clasificador y la evaluación del modelo de su rendimiento.

3.2.1 Configuración Inicial y Gestión de Archivos

Como se muestra en la Ilustración 17 en esta sección establece el entorno de ejecución. Este entorno se refiere a Python, el cual se configura mediante la importación de las librerías necesarias y definiendo las rutas de los directorios donde se almacenará el modelo. Se implementa una fase de limpieza para asegurar que cada proceso de entrenamiento comience en un estado limpio, eliminando versiones anteriores de los modelos.

Ilustración 17

Configuración



```python
import os
import sys
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import classification_report, confusion_matrix, roc_curve, auc, precision_recall_curve, average_precision_score, accuracy_score, precision_score, recall_sco
from joblib import dump, load
from pathlib import Path
import math
import pyarrow.parquet as pq
import shutil
import logging

# --- Configuración de Logging ---
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

# --- Rutas de Archivos ---
SCRIPT_DIR = Path(__file__).parent
PROCESSED_FILES_FOLDER = SCRIPT_DIR.parent / "processed_files"
ML_MODELS_DIR = SCRIPT_DIR

ML_MODELS_DIR.mkdir(parents=True, exist_ok=True)
logging.info(f"Carpeta de modelos ML asegurada: {ML_MODELS_DIR}")

# --- Limpiar el directorio de modelos antes de entrenar ---
print(f"Limpiando contenido de la carpeta de modelos ML '{ML_MODELS_DIR}' antes de la regeneración...")
for item in ML_MODELS_DIR.iterdir():
    if item.is_file() and (item.suffix == '.pkl' or item.suffix == '.bin'):
        try:
            item.unlink()
            logging.info(f"Eliminado archivo antiguo: {item.name}")
        except OSError as e:
            logging.warning(f"No se pudo eliminar el archivo {item.name}: {e}")
    elif item.is_dir():
        pass
logging.info(f"Limpieza de archivos .pkl y .bin en '{ML_MODELS_DIR}' completada.")
```

Se importaron librerías fundamentales para Machine Learning (sklearn), manejo de datos (pandas, numpy, PyArrow. parquet), sistema de archivos (os, pathlib, shutil) y registro de eventos (logging). Se definen rutas absolutas para asegurar la portabilidad del script, creando o verificando la existencia del directorio ML_MODELS_DIR. Un bucle iterativo limpia estos directorios de artefactos de modelo previos (. pkl, bin), garantizando que cada entrenamiento sea producible y no haya conflicto con versiones antiguas de modelos.

3.2.2 Definición de Características y Configuración de Procesamiento

En la Ilustración 30 se indica que esta sección se especifica las columnas del Dataset que serán utilizadas por el modelo. Además, se establecen parámetros clave para optimizar el manejo de grandes volúmenes de datos durante el entrenamiento.

Ilustración 18

Características

```
# --- Definiciones de Columnas
COLUMNS_FOR_MODEL_NUMERIC = [
    'bwd_iat_total', 'bwd_iat_std', 'flow_byts_s', 'bwd_pkt_len_std', 'fwd_pkt_len_mean',
    'fwd_psh_flags', 'active_min', 'fwd_pkts_b_avg', 'pkt_len_var',
    'fwd_pkt_len_std', 'totlen_fwd_pkts', 'bwd_bulk_rate_avg', 'init_bwd_win_byts',
    'fwd_bulk_rate_avg', 'fwd_pkt_len_max', 'flow_iat_std', 'cwe_flag_count',
    'subflow_fwd_pkts', 'fwd_urg_flags', 'syn_flag_cnt', 'tot_fwd_pkts',
    'subflow_bwd_pkts', 'bwd_byts_b_avg', 'act_data_pkt_fwd', 'bwd_pkts_b_avg',
    'subflow_bwd_byts', 'bwd_iat_mean', 'bwd_pkt_len_max', 'down_up_ratio',
    'bwd_header_len', 'init_fwd_win_byts', 'bwd_iat_min', 'fwd_iat_max',
    'bwd_urg_flags', 'bwd_psh_flags', 'idle_std', 'idle_max', 'fwd_byts_b_avg',
    'pkt_len_std', 'active_std', 'totlen_bwd_pkts',
    'tot_bwd_pkts',
    'flow_iat_mean', 'min_seg_size_fwd', 'fwd_iat_mean', 'pkt_size_avg',
    'active_max', 'fwd_iat_total', 'ece_flag_cnt', 'fwd_iat_std',
    'fwd_header_len', 'max_pkt_len', 'rst_flag_cnt', 'flow_iat_min',
    'bwd_seg_size_avg', 'pkt_len_mean', 'idle_mean', 'idle_min', 'fin_flag_cnt',
    'psh_flag_cnt', 'fwd_seg_size_avg', 'bwd_pkt_len_mean', 'min_pkt_len',
    'flow_iat_max', 'ack_flag_cnt', 'fwd_pkt_len_min', 'bwd_pkt_len_min',
    'flow_duration', 'flow_pkts_s', 'urg_flag_cnt', 'fwd_iat_min',
    'active_mean', 'subflow_fwd_byts', 'bwd_pkts_s', 'fwd_header_len1', 'bwd_iat_max'
]
PROTOCOL_COLUMN = 'protocol'
LABEL_COLUMN = 'label'

ALL_REQUIRED_COLUMNS_FOR_LOADING = COLUMNS_FOR_MODEL_NUMERIC + [PROTOCOL_COLUMN, LABEL_COLUMN]

# --- Configuración de Muestreo y Lotes ---
MAX_TRAINING_SAMPLES = 1_000_000
SAMPLING_RANDOM_STATE = 42
BATCH_SIZE_SCALER = 100_000
```

Se definen listas de cadenas (COLUMNS_FOR_MODEL_NUMERIC) que corresponden a las características numéricas extraídas del tráfico de red que el modelo utilizará para la clasificación. La inclusión de PROTOCOL_COLUMN y LABEL_COLUMN es fundamental para características categóricas y la variable objetivo, respectivamente. Parámetros como MAX_TRAINING_SAMPLES y BATCH_SIZE_SCALER reflejan una estrategia de optimización para manejar grandes datasets, permitiendo el procesamiento en lotes y el muestreo para eficiencia de memoria y tiempo.

3.2.3 Proceso de entrenamiento y guardar del modelo

En la Ilustración 31 se muestra un fragmento de código la cuál esta es la función principal de entrenamiento. Su diseño está optimizado para trabajar con grandes volúmenes de datos almacenados en formato Parquet, gestionando la memoria de forma eficiente.

- El Label Enconder es fundamental para el procesamiento de la variante objetivo, esto se encarga de la codificación de las etiquetas de clase (´BENIGN´, ´MSSQL´).

Ilustración 19

Entrenamiento del modelo

```
84
85      # 1. Ajustar LabelEncoder: Recopilar todas las etiquetas únicas de todos los archivos Parquet
86      all_unique_labels = set()
87      print("Recolectando etiquetas únicas para LabelEncoder...")
88      for pq_file in parquet_files:
89          try:
90              df_labels = pd.read_parquet(pq_file, columns=[LABEL_COLUMN])
91              all_unique_labels.update(df_labels[LABEL_COLUMN].astype(str).unique())
92          except Exception as e:
93              logging.warning(f"No se pudieron leer las etiquetas de {pq_file.name}: {e}. Se omitirá este archivo para las etiquetas.", exc_info=True)
94
95      if not all_unique_labels:
96          print("❌ Error: No se encontraron etiquetas válidas en ningún archivo Parquet. No se puede entrenar el modelo.")
97          sys.exit(1)
98
99      label_encoder = LabelEncoder()
100     sorted_unique_labels = sorted(list(all_unique_labels))
101     label_encoder.fit(sorted_unique_labels)
102     print(f"✅ LabelEncoder ajustado con {len(label_encoder.classes_)} clases: {label_encoder.classes_}")
103
```

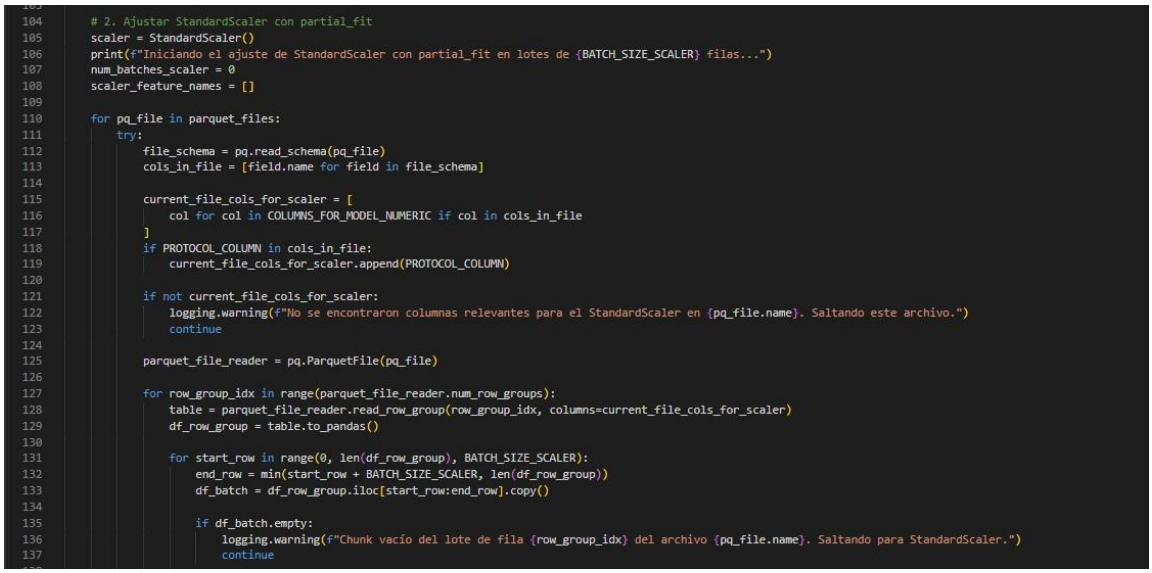
Para asegurar que todas las posibles clases de tráfico (benigno y distintos tipos de ataques) sean correctamente mapeados a valores numéricos, se inicializa un LabelEncoder. Se ajusta globalmente a partir de la recopilación de todas las etiquetas únicas presentes en el conjunto completo de archivos Parquet. Esto evita problemas de "clases no vistas", durante la predicción y asegura una codificación consistente.

3.2.4 Ajuste del StandarScaler con partial_fit (Optimización para Big data)

La estandarización de las características numéricas es un paso fundamental en muchos modelos de aprendizaje automático, incluyendo Random Forest, aunque resulta aún más relevante en aquellos algoritmos que se basan en distancias. Este proceso se hizo mediante la instanciación de un StandarScaler de Scikit-learn, el cual se ajustó de manera incremental utilizando el método partial_fit.

Ilustración 20

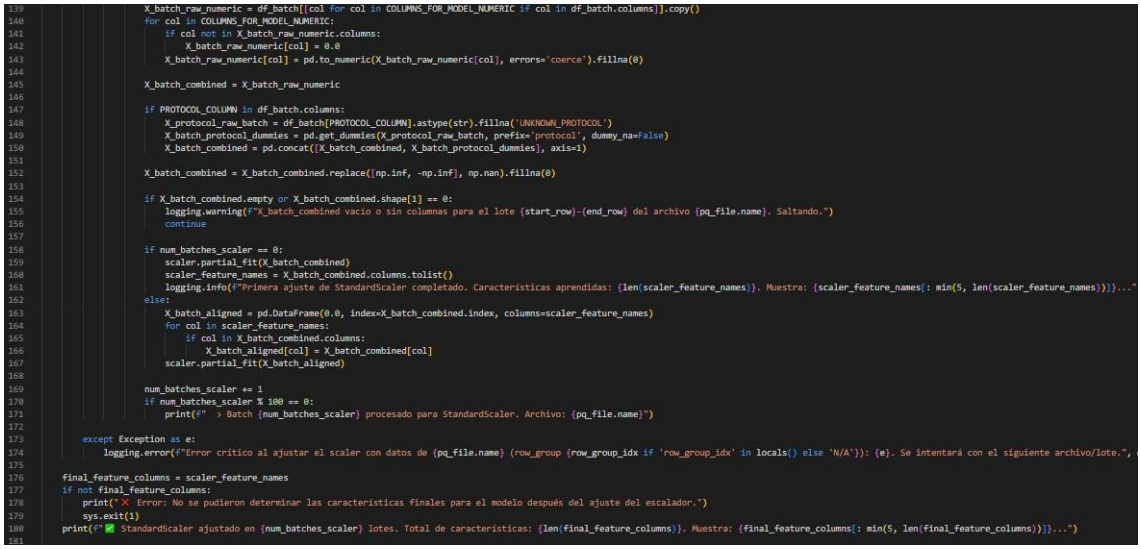
Ajuste StandarScaler



```
    # 2. Ajustar StandardScaler con partial_fit
    scaler = StandardScaler()
    print(f"Iniciando el ajuste de StandardScaler con partial_fit en lotes de {BATCH_SIZE_SCALER} filas...")
    num_batches_scaler = 0
    scaler_feature_names = []

    for pq_file in parquet_files:
        try:
            file_schema = pq.read_schema(pq_file)
            cols_in_file = [field.name for field in file_schema]

            current_file_cols_for_scaler = [
                col for col in COLUMNS_FOR_MODEL_NUMERIC if col in cols_in_file
            ]
            if PROTOCOL_COLUMN in cols_in_file:
                current_file_cols_for_scaler.append(PROTOCOL_COLUMN)

            if not current_file_cols_for_scaler:
                logging.warning(f"No se encontraron columnas relevantes para el StandardScaler en {pq_file.name}. Saltando este archivo.")
                continue

            parquet_file_reader = pq.ParquetFile(pq_file)

            for row_group_idx in range(parquet_file_reader.num_row_groups):
                table = parquet_file_reader.read_row_group(row_group_idx, columns=current_file_cols_for_scaler)
                df_row_group = table.to_pandas()

                for start_row in range(0, len(df_row_group), BATCH_SIZE_SCALER):
                    end_row = min(start_row + BATCH_SIZE_SCALER, len(df_row_group))
                    df_batch = df_row_group.iloc[start_row:end_row].copy()

                    if df_batch.empty:
                        logging.warning(f"Chunk vacío del lote de fila {row_group_idx} del archivo {pq_file.name}. Saltando para StandardScaler.")
                        continue
```

Ilustración 21

Ajuste StandarScaler



```
                    X_batch_raw_numeric = df_batch[[col for col in COLUMNS_FOR_MODEL_NUMERIC if col in df_batch.columns]].copy()
                    for col in COLUMNS_FOR_MODEL_NUMERIC:
                        if col not in X_batch_raw_numeric.columns:
                            X_batch_raw_numeric[col] = 0.0
                        X_batch_raw_numeric[col] = pd.to_numeric(X_batch_raw_numeric[col], errors='coerce').fillna(0)

                    X_batch_combined = X_batch_raw_numeric

                    if PROTOCOL_COLUMN in df_batch.columns:
                        X_protocol_raw_batch = df_batch[PROTOCOL_COLUMN].astype(str).fillna('UNKNOWN_PROTOCOL')
                        X_batch_protocol_dummies = pd.get_dummies(X_protocol_raw_batch, prefix='protocol', dummy_na=False)
                        X_batch_combined = pd.concat([X_batch_combined, X_batch_protocol_dummies], axis=1)

                    X_batch_combined = X_batch_combined.replace([np.inf, -np.inf], np.nan).fillna(0)

                    if X_batch_combined.empty or X_batch_combined.shape[1] == 0:
                        logging.warning(f"X_batch_combined vacío o sin columnas para el lote {start_row}-{end_row} del archivo {pq_file.name}. Saltando.")
                        continue

                    if num_batches_scaler == 0:
                        scaler.partial_fit(X_batch_combined)
                        scaler_feature_names = X_batch_combined.columns.tolist()
                        logging.info(f"Primera ajuste de StandardScaler completado. Características aprendidas: {len(scaler_feature_names)}. Muestra: {scaler_feature_names[: min(5, len(scaler_feature_names))]}...")
                    else:
                        X_batch_aligned = pd.DataFrame(0.0, index=X_batch_combined.index, columns=scaler_feature_names)
                        for col in scaler_feature_names:
                            if col in X_batch_combined.columns:
                                X_batch_aligned[col] = X_batch_combined[col]
                        scaler.partial_fit(X_batch_aligned)

                    num_batches_scaler += 1
                    if num_batches_scaler % 100 == 0:
                        print(f"  > Batch {num_batches_scaler} procesado para StandardScaler. Archivo: {pq_file.name}")

        except Exception as e:
            logging.error(f"Error crítico al ajustar el scaler con datos de {pq_file.name} (row_group {row_group_idx if 'row_group_idx' in locals() else 'N/A'}): {e}. Se intentará con el siguiente archivo/lote.",

    final_feature_columns = scaler_feature_names
    if not final_feature_columns:
        print("❌ Error: No se pudieron determinar las características finales para el modelo después del ajuste del escalador.")
        sys.exit(1)
    print(f"✅ StandardScaler ajustado en {num_batches_scaler} lotes. Total de características: {len(final_feature_columns)}. Muestra: {final_feature_columns[: min(5, len(final_feature_columns))]}...")
```

Se utilizó un StandardScaler para normalizar las características numéricas, ajustándolas para que tengan una media igual a cero y una desviación estándar igual a uno, lo que facilita el desempeño de los modelos de aprendizaje automático. Esto es crucial para asegurar que ninguna característica domine el proceso de aprendizaje debido a su escala. La implementación utiliza partial_fit en lotes (BATCH_SIZE_SCALER). Este enfoque es una técnica de aprendizaje incremental, que permite al escalador aprender las estadísticas (media y desviación estándar) de un Dataset que no cabe completamente en memoria RAM. Para las características categóricas, se realiza un One-Hot Encoding

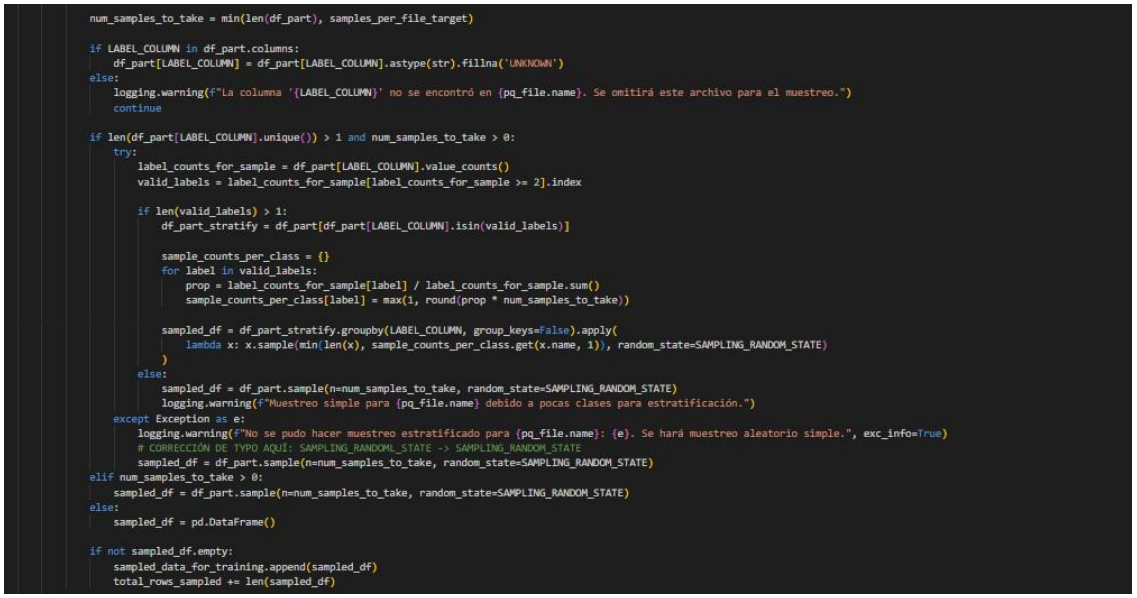
(pd.get_dummies) para convertirlas en un formato numérico adecuado para el modelo. Se maneja valores NaN, inf y -inf para asegurar la robustez de los datos. La consistencia de las columnas (scaler_feature_names) es mantenida a lo largo de los lotes para asegurar que el scaler se ajuste correctamente a todas las dimensiones.

3.2.5 Recopilación de Datos Muestreados para el Entrenamiento

Dado el volumen potencialmente grande de los datos de tráfico, se implementa una estrategia de muestreo para optimizar

Ilustración 22

Recopilación



```
num_samples_to_take = min(len(df_part), samples_per_file_target)

if LABEL_COLUMN in df_part.columns:
    df_part[LABEL_COLUMN] = df_part[LABEL_COLUMN].astype(str).fillna('UNKNOWN')
else:
    logging.warning(f"La columna '{LABEL_COLUMN}' no se encontró en {pq_file.name}. Se omitirá este archivo para el muestreo.")
    continue

if len(df_part[LABEL_COLUMN].unique()) > 1 and num_samples_to_take > 0:
    try:
        label_counts_for_sample = df_part[LABEL_COLUMN].value_counts()
        valid_labels = label_counts_for_sample[label_counts_for_sample >= 2].index

        if len(valid_labels) > 1:
            df_part_stratify = df_part[df_part[LABEL_COLUMN].isin(valid_labels)]

            sample_counts_per_class = {}
            for label in valid_labels:
                prop = label_counts_for_sample[label] / label_counts_for_sample.sum()
                sample_counts_per_class[label] = max(1, round(prop * num_samples_to_take))

            sampled_df = df_part_stratify.groupby(LABEL_COLUMN, group_keys=False).apply(
                lambda x: x.sample(min(len(x), sample_counts_per_class.get(x.name, 1)), random_state=SAMPLING_RANDOM_STATE)
            )
        else:
            sampled_df = df_part.sample(n=num_samples_to_take, random_state=SAMPLING_RANDOM_STATE)
            logging.warning(f"Muestreo simple para {pq_file.name} debido a pocas clases para estratificación.")
    except Exception as e:
        logging.warning(f"No se pudo hacer muestreo estratificado para {pq_file.name}: {e}. Se hará muestreo aleatorio simple.", exc_info=True)
        # CORRECCIÓN DE TYPO AQUÍ: SAMPLING_RANDOML_STATE -> SAMPLING_RANDOM_STATE
        sampled_df = df_part.sample(n=num_samples_to_take, random_state=SAMPLING_RANDOM_STATE)
elif num_samples_to_take > 0:
    sampled_df = df_part.sample(n=num_samples_to_take, random_state=SAMPLING_RANDOM_STATE)
else:
    sampled_df = pd.DataFrame()

if not sampled_df.empty:
    sampled_data_for_training.append(sampled_df)
    total_rows_sampled += len(sampled_df)
```

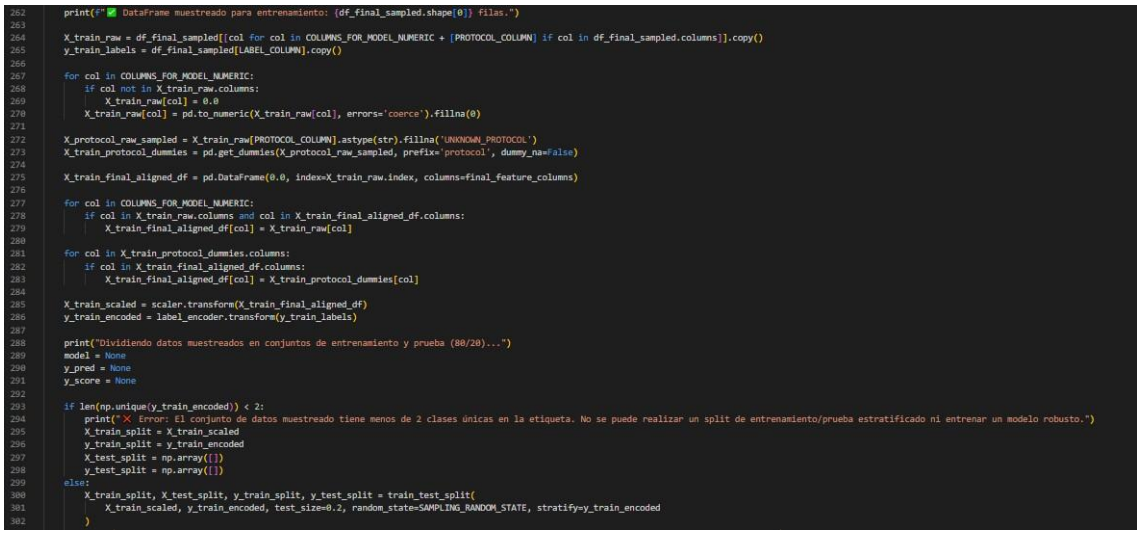
El entrenamiento se realiza sobre una muestra representativa del conjunto de datos completo, limitada por el parámetro MAX_TRAINING_SAMPLES. Para esto, se calcula un numero objetivo de muestras por archivo con el fin de distribuir el muestreo de manera uniforme. Lo que favorece el muestreo estratificado, los grupos para realizar el muestreo pertenecen a las diferentes clases presentes en la columna (label), como ´Benign´, ´MSSQL´, ´LDAP´.

3.2.6 Preparación final de los datos y División en Conjuntos de Entrenamiento/Prueba

Al entrenar del modelo, los datos muestreados se transforman y se dividen para una evaluación rigurosa.

Ilustración 23

Preparación y División



```
    print(f"✅ DataFrame muestreado para entrenamiento: {df_final_sampled.shape[0]} filas.")

    X_train_raw = df_final_sampled[[col for col in COLUMNS_FOR_MODEL_NUMERIC + [PROTOCOL_COLUMN] if col in df_final_sampled.columns]].copy()
    y_train_labels = df_final_sampled[LABEL_COLUMN].copy()

    for col in COLUMNS_FOR_MODEL_NUMERIC:
        if col not in X_train_raw.columns:
            X_train_raw[col] = 0.0
        X_train_raw[col] = pd.to_numeric(X_train_raw[col], errors='coerce').fillna(0)

    X_protocol_raw_sampled = X_train_raw[PROTOCOL_COLUMN].astype(str).fillna('UNKNOWN_PROTOCOL')
    X_train_protocol_dummies = pd.get_dummies(X_protocol_raw_sampled, prefix='protocol', dummy_na=False)

    X_train_final_aligned_df = pd.DataFrame(0.0, index=X_train_raw.index, columns=final_feature_columns)

    for col in COLUMNS_FOR_MODEL_NUMERIC:
        if col in X_train_raw.columns and col in X_train_final_aligned_df.columns:
            X_train_final_aligned_df[col] = X_train_raw[col]

    for col in X_train_protocol_dummies.columns:
        if col in X_train_final_aligned_df.columns:
            X_train_final_aligned_df[col] = X_train_protocol_dummies[col]

    X_train_scaled = scaler.transform(X_train_final_aligned_df)
    y_train_encoded = label_encoder.transform(y_train_labels)

    print("Dividiendo datos muestreados en conjuntos de entrenamiento y prueba (80/20)...")
    model = None
    y_pred = None
    y_score = None

    if len(np.unique(y_train_encoded)) < 2:
        print("❌ Error: El conjunto de datos muestreado tiene menos de 2 clases únicas en la etiqueta. No se puede realizar un split de entrenamiento/prueba estratificado ni entrenar un modelo robusto.")
        X_train_split = X_train_scaled
        y_train_split = y_train_encoded
        X_test_split = np.array([])
        y_test_split = np.array([])
    else:
        X_train_split, X_test_split, y_train_split, y_test_split = train_test_split(
            X_train_scaled, y_train_encoded, test_size=0.2, random_state=SAMPLING_RANDOM_STATE, stratify=y_train_encoded
        )
```

Las muestras obtenidas se combinan en único DataFrame, del cual se separan las características (X_train_raw) y las etiquetas (y_train_labels). Es fundamental convertir las características numéricas a tipos adecuados y reemplazar los valores inválidos, como NaN o infinitos. Las variables categóricas de protocolo se codifican mediante One-Hot Encoding, generando columnas binarias adicionales. Posteriormente se asegura que las columnas de X_train_raw coincidan exactamente en orden y cantidad con las características esperadas por el StandardScaler, incluyendo las variables dummy de protocolo.

3.2.7 Entramiento del modelo RandomForestClassifier

Es el corazón sistema de detección es el modelo de Machine Learning, en este caso, RandomForestClassifier.

Ilustración 24

Entrenamiento del modelo

```
    print("Entrenando RandomForestClassifier con dato  (parameter) random_state: Int | RandomState | None
    model = RandomForestClassifier(n_estimators=100, random_state=SAMPLING_RANDOM_STATE, n_jobs=-1, verbose=1,
                                   class_weight='balanced')
    model.fit(X_train_split, y_train_split)
    print("✅ Entrenamiento del modelo completado.")

print("\n--- Evaluación del Modelo en el Conjunto de Prueba Muestreado ---")
```

Los parámetros clave son:

- n_estimators=100: Define el número de árboles de decisión en el bosque. Un mayor número generalmente mejora la robustez.

- random_state=SAMPLING_RANDOM_STATE: Asegura la reproducibilidad de los resultados.
- N_jobs=1 Habilita la salida de progreso durante el entrenamiento.
- Class_weight=`Balanced`: Este parámetro es fundamental para conjuntos de datos, ya que se ajusta automáticamente los pesos de cada clase en función inversa a su frecuencia dentro del conjunto de entrenamiento

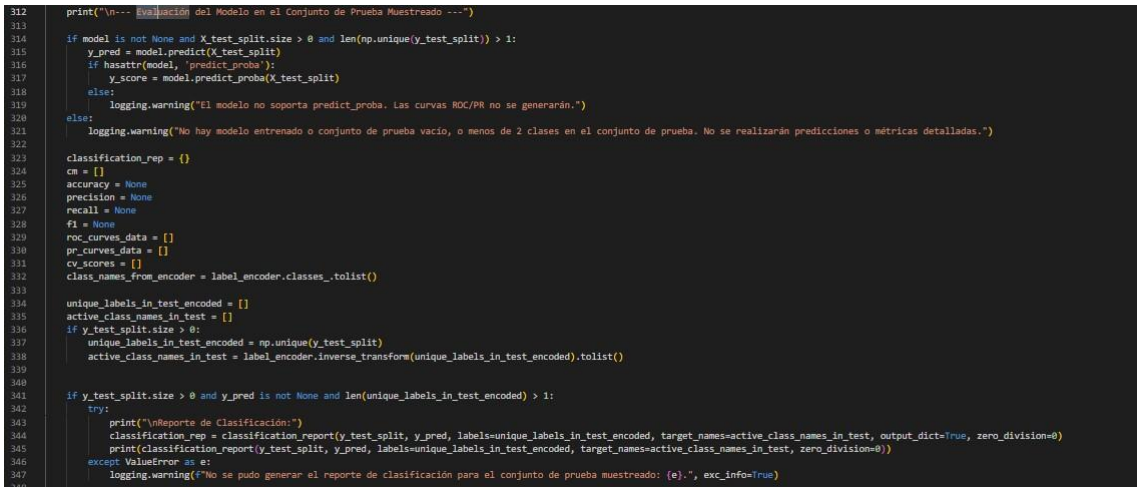
El método fit () entrena el modelo utilizando el conjunto de entrenamiento escalado y las etiquetas codificadas.

3.2.8 Evaluación del Modelo y Cálculo de Métricas de Rendimiento

En las Ilustraciones 37 y 38 se muestra el modelo ya entrenado, el modelo se evalúa rigorosamente utilizando el conjunto de prueba para determinar su capacidad.

Ilustración 25

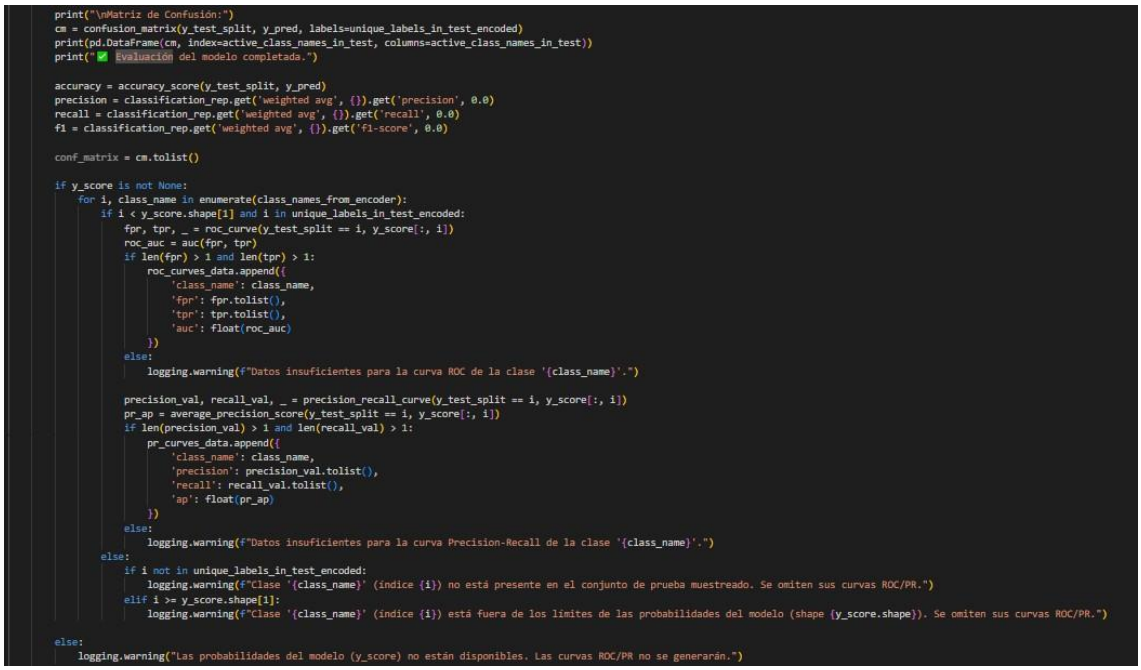
Evaluación y Cálculo



```
print("\n--- Evaluación del Modelo en el Conjunto de Prueba Muestreado ---")

if model is not None and X_test_split.size > 0 and len(np.unique(y_test_split)) > 1:
    y_pred = model.predict(X_test_split)
    if hasattr(model, 'predict_proba'):
        y_score = model.predict_proba(X_test_split)
    else:
        logging.warning("El modelo no soporta predict_proba. Las curvas ROC/PR no se generarán.")
else:
    logging.warning("No hay modelo entrenado o conjunto de prueba vacío, o menos de 2 clases en el conjunto de prueba. No se realizarán predicciones o métricas detalladas.")

classification_rep = {}
cm = []
accuracy = None
precision = None
recall = None
f1 = None
roc_curves_data = []
pr_curves_data = []
cv_scores = []
class_names_from_encoder = label_encoder.classes_.tolist()

unique_labels_in_test_encoded = []
active_class_names_in_test = []
if y_test_split.size > 0:
    unique_labels_in_test_encoded = np.unique(y_test_split)
    active_class_names_in_test = label_encoder.inverse_transform(unique_labels_in_test_encoded).tolist()


if y_test_split.size > 0 and y_pred is not None and len(unique_labels_in_test_encoded) > 1:
    try:
        print("\nReporte de Clasificación:")
        classification_rep = classification_report(y_test_split, y_pred, labels=unique_labels_in_test_encoded, target_names=active_class_names_in_test, output_dict=True, zero_division=0)
        print(classification_report(y_test_split, y_pred, labels=unique_labels_in_test_encoded, target_names=active_class_names_in_test, zero_division=0))
    except ValueError as e:
        logging.warning(f"No se pudo generar el reporte de clasificación para el conjunto de prueba muestreado: {e}.", exc_info=True)
```

Ilustración 26

Evaluación y Cálculo



```
print("\nMatriz de Confusión:")
cm = confusion_matrix(y_test_split, y_pred, labels=unique_labels_in_test_encoded)
print(pd.DataFrame(cm, index=active_class_names_in_test, columns=active_class_names_in_test))
print("✅ Evaluación del modelo completada.")

accuracy = accuracy_score(y_test_split, y_pred)
precision = classification_rep.get('weighted avg', {}).get('precision', 0.0)
recall = classification_rep.get('weighted avg', {}).get('recall', 0.0)
f1 = classification_rep.get('weighted avg', {}).get('f1-score', 0.0)

conf_matrix = cm.tolist()

if y_score is not None:
    for i, class_name in enumerate(class_names_from_encoder):
        if i < y_score.shape[1] and i in unique_labels_in_test_encoded:
            fpr, tpr, _ = roc_curve(y_test_split == i, y_score[:, i])
            roc_auc = auc(fpr, tpr)
            if len(fpr) > 1 and len(tpr) > 1:
                roc_curves_data.append({
                    'class_name': class_name,
                    'fpr': fpr.tolist(),
                    'tpr': tpr.tolist(),
                    'auc': float(roc_auc)
                })
            else:
                logging.warning(f"Datos insuficientes para la curva ROC de la clase '{class_name}'.")

            precision_val, recall_val, _ = precision_recall_curve(y_test_split == i, y_score[:, i])
            pr_ap = average_precision_score(y_test_split == i, y_score[:, i])
            if len(precision_val) > 1 and len(recall_val) > 1:
                pr_curves_data.append({
                    'class_name': class_name,
                    'precision': precision_val.tolist(),
                    'recall': recall_val.tolist(),
                    'ap': float(pr_ap)
                })
            else:
                logging.warning(f"Datos insuficientes para la curva Precision-Recall de la clase '{class_name}'.")
        else:
            if i not in unique_labels_in_test_encoded:
                logging.warning(f"Clase '{class_name}' (índice {i}) no está presente en el conjunto de prueba muestreado. Se omiten sus curvas ROC/PR.")
            elif i >= y_score.shape[1]:
                logging.warning(f"Clase '{class_name}' (índice {i}) está fuera de los límites de las probabilidades del modelo (shape {y_score.shape}). Se omiten sus curvas ROC/PR.")

else:
    logging.warning("Las probabilidades del modelo (y_score) no están disponibles. Las curvas ROC/PR no se generarán.")
```

El modelo genera predicciones de etiqueta (y_pred) y, cuando es posible, también calcula las probabilidades de cada (y_score) para el conjunto de pruebas.

- Classification_report: Ofrece medidas detalladas como precisión, exhaustividad o recall, F1-Score y soporte para cada clase.
- Confusión_matrix: Presenta la cantidad de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos, lo que permite realizar un análisis detallado.
- Accuracy_score, precision_score, recall_score y f1_score: Proporciona métricas globales que resumen el rendimiento general del modelo.
- Curva ROC: Representa la tasa de verdaderos positivos frente a la tasa de falsos positivos para distintos umbrales.
- Curva Precision-Recall: Muestra la relación entre precisión y exhaustividad, siendo especialmente útil en conjuntos de datos desequilibrados.

Los resultados de evaluación se obtuvieron mediante validación cruzada, lo que garantizo la fiabilidad en un sistema crítico. Esta confirmó que el modelo Random Forest presenta un rendimiento consistente, con una desviación estándar inferior a 0.05 en los cinco pliegues, lo que valida tanto su estabilidad como su capacidad de generalización frente a nuevos ataques DoS en redes Wi-Fi.

3.2.9 Guardado del Modelo

El paso final es la persistencia de todos los componentes para desplegar y utilizar el modelo.

Ilustración 27

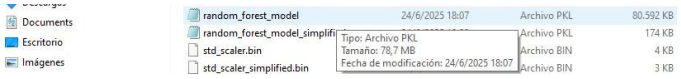
Guardado del modelo

```
print("\nGuardando el modelo, el scaler, el label encoder, la lista de caracteristicas finales y las métricas...")
if model and scaler and label_encoder and final_feature_columns:
    dump(model, ML_MODELS_DIR / "random_forest_model.pkl")
    dump(scaler, ML_MODELS_DIR / "std_scaler.bin")
    dump(label_encoder, ML_MODELS_DIR / "label_encoder.bin")
    dump(final_feature_columns, ML_MODELS_DIR / "model_features.pkl")
    dump(model_metrics_to_save, ML_MODELS_DIR / "model_metrics.pkl")
    print("✅ Componentes del modelo guardados exitosamente.")
```

- Random_forest_model.pkl: El modelo RandomForestClassifier entrenado

Ilustración 28

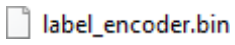
Random_forest_model.pkl



- Label_encoder.bin: El LabelEncoder ajustado, necesario para decodificar las predicciones del modelo de vuelta a sus etiquetas originales.

Ilustración 29

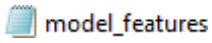
Label_encoder.bin



- Model_feactures.pkl: Asegura que los datos de entrada para la predicción tengan el orden y el número exacto de características.

Ilustración 30

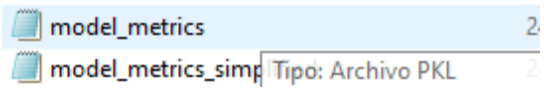
model_features



- Model_metrics.pkl: Contiene todas las métricas de evaluación calculadas.

Ilustración 31

Model_metrics.pkl



3.3 Resultados fase 2

Una vez finalizada la Fase 1, se procedió al diseño de un sistema web orientado a la detección de ataques DoS en redes Wi-Fi. En esta fase se tuvo como objetivo el diseño interactivo de la interfaz y funcional para la detección temprana de ataques.

Durante esta fase, se optimizó el manejo de datos transformando los archivos CSV, generados a partir de capturas de tráfico, al formato **Parquet**, lo que permitió una carga más eficiente dentro del sistema. Para facilitar la accesibilidad en la detección de ataques y posibles amenazas, se determinó que el sistema no requerirá autenticación del usuario.

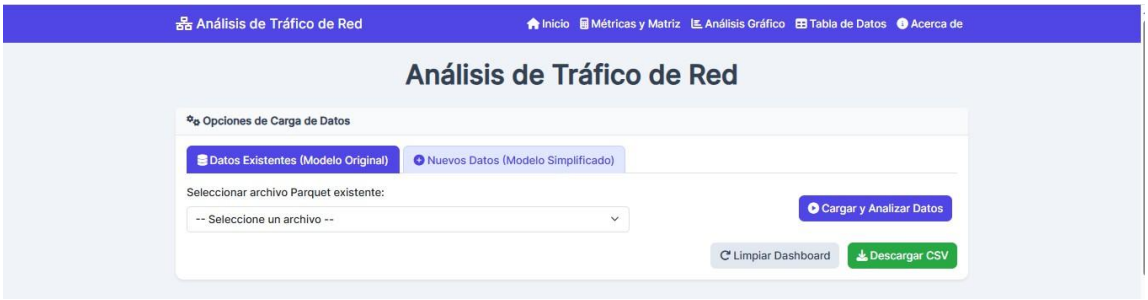
3.3.1 Módulo de gestión de archivos de tráfico (Interfaz Principal)

En la Ilustración 17 se muestra una interfaz donde el usuario puede seleccionar, cargar y analizar archivos de tráfico en formato. parquet.

Desde la interfaz, se muestra un listado con los archivos para poder cargarlos. A través de esta vista el usuario puede seleccionarlos para un análisis como se observa en las Ilustración 18. Entre las acciones disponibles se encuentran: cargar y analizar Datos, Limpiar Dashboard, Descargar CSV.

Ilustración 32

Selección de Archivos



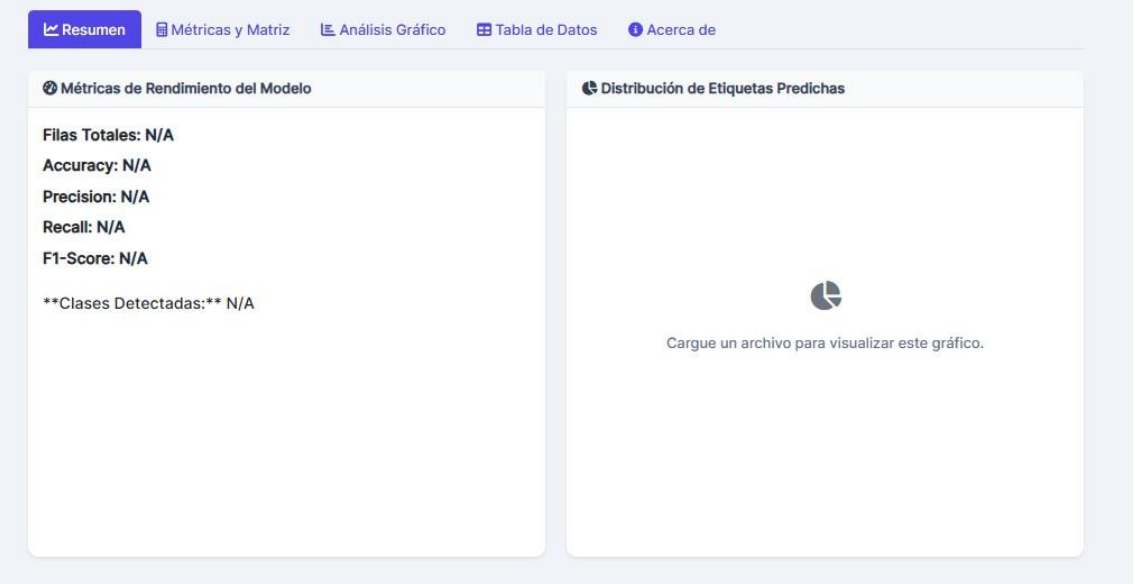
El sistema web se estructura en varias pestañas, cada una presenta una visualización de diferentes aspectos del análisis. A continuación, se presentan vistas que contiene cada pestaña, indicando al usuario que debe cargar archivos para visualizar los resultados correspondientes.

- **Pestaña "Resumen":** La Ilustración 18 muestra la pestaña Resumen donde se proporciona métricas generales del rendimiento del modelo, como Accuary, precisión, recall y F1-Score, junto a una gráfica de pastel representada por la

distribución de clases predichas en el conjunto cargado. En la vista inicial, se indica claramente que no hay métricas disponibles y se requiere la carga de archivos para visualizar el tráfico.

Ilustración 33

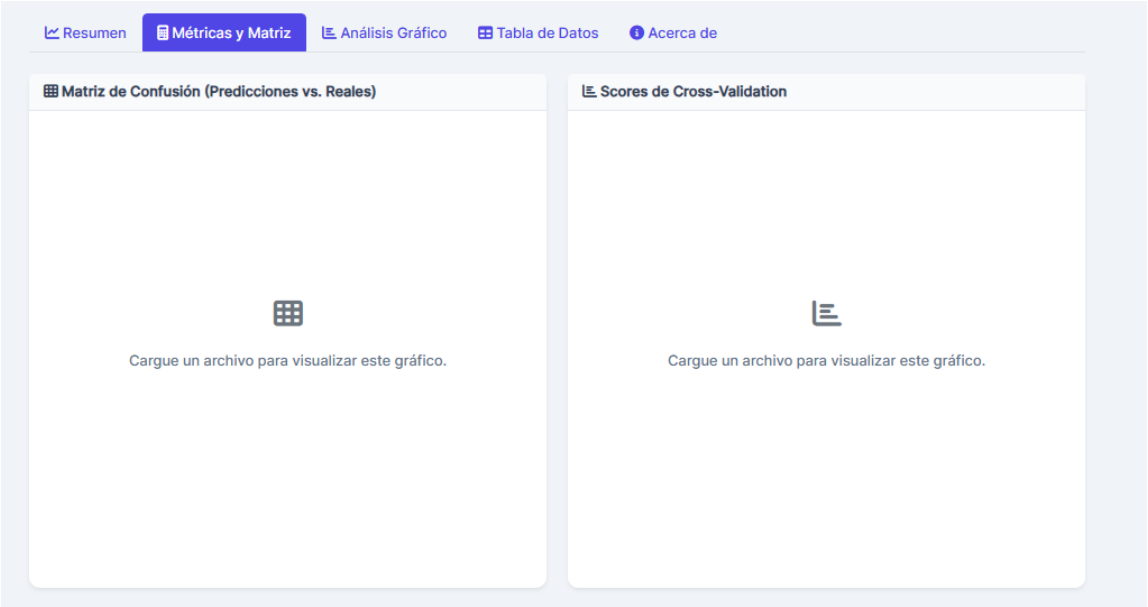
Resumen



- **Pestaña Métricas y Matriz:** La Ilustración 34 corresponde a la pestaña "Métricas y Matriz". En esta sección se muestra la matriz de confusión, que detalla las predicciones del modelo frente a los valores reales, también los scores de validación cruzada. En este estado, el sistema solicita al usuario que cargue un archivo para generar y visualizar gráficos, que son esenciales para una evaluación profunda del rendimiento del modelo. El sistema requiere que se cargue un archivo.

Ilustración 34

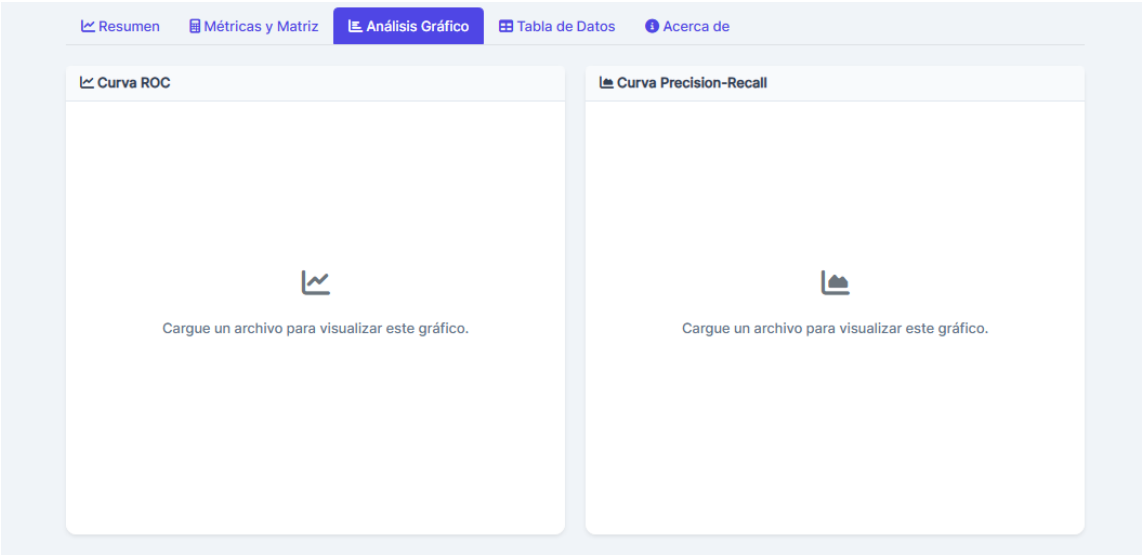
Métricas y Matriz



- **Pestaña "Análisis Gráfico":** En la Ilustración 35 se muestra la pestaña "Análisis Gráfico" en su estado inicial. Esta sección está diseñada para visualizar las curvas ROC (Receiver Operating Characteristic) y Precision-Recall, lo cual son fundamentales para poder evaluar el modelo. Como se observa, el sistema requiere la carga de un archivo para generar las curvas.

Ilustración 35

Análisis Gráfico

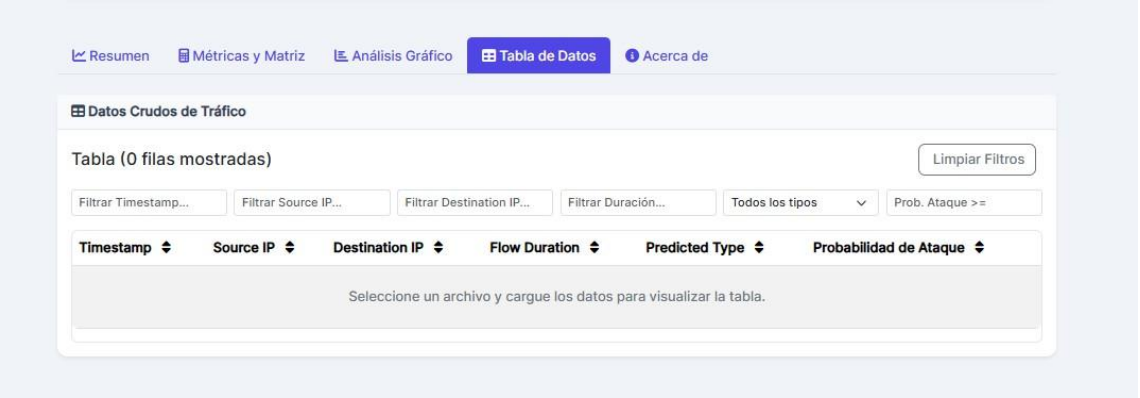


- **Pestaña "Tabla de Datos":** La Ilustración 36 presenta la pestaña "Tabla de Datos" en su estado predeterminado. Este módulo está diseñado para mostrar los

datos de tráfico que se hayan procesado, por el modelo, incluyendo times, IPs de origen y destino, duración de flujo, tipo predicho, y probabilidad de ataque. En la vista inicial se encuentra vacía, que indica al usuario cargar el archivo para visualizar los datos.

Ilustración 36

Pestaña Tabla de Datos



3.3.2 Módulo del historial de análisis (Pestaña "Tabla de Datos")

El sistema desarrollado incorpora un **Módulo del Historial de Análisis,** accesible a través de la pestaña "Tabla de Datos". Este módulo es fundamental para la visualización detallada de los registros de tráfico de red procesados por el modelo. Permite al usuario visualizar los datos del tráfico después de que han sido cargados y analizados, proporcionando así una inspección granular de las características y las predicciones del modelo.

Lo que también es primordial e importante es que este módulo gestiona los datos: cada vez que se carga un archivo parquet para su análisis. De este modo, la tabla siempre muestra el conjunto de datos y los resultados correspondientes a la última ejecución del modelo.

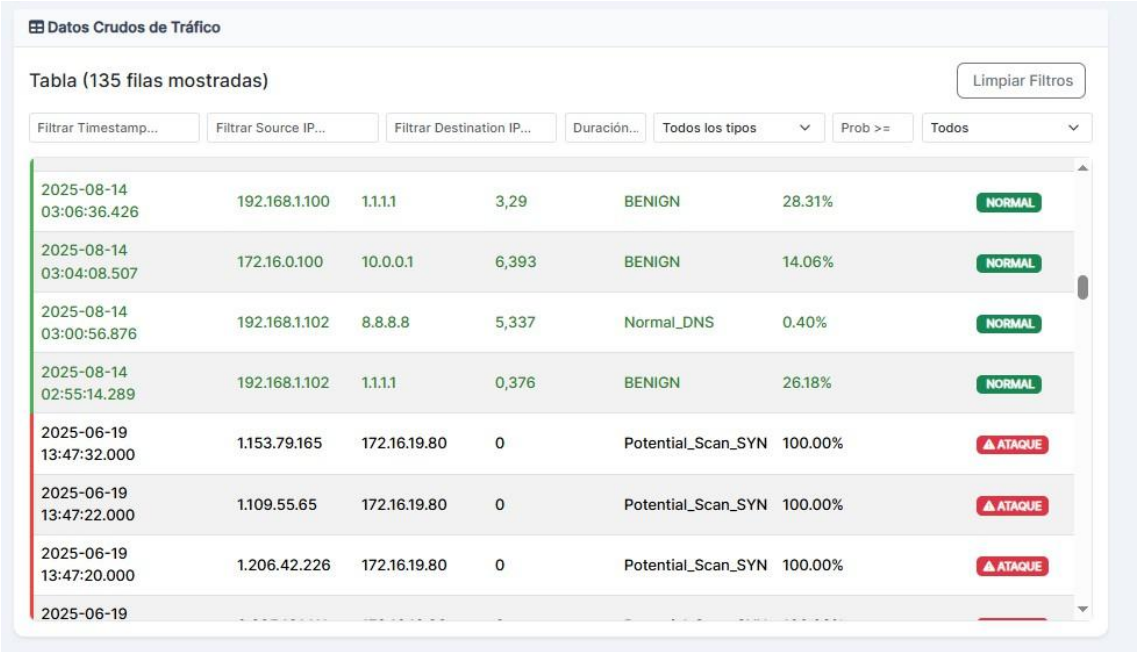
En la Ilustración 37 se observa una organización y el contenido del módulo, donde cada columna aporta información relevante sobre los flujos de tráfico analizados y las inferencias generadas por el modelo:

- **Timestamp:** Representa la fecha y la hora en la se registró el inicio de un flujo de tráfico de red. Es crucial para el análisis cronológico y la correlación de eventos de seguridad. Permite rastrear cuando ocurrió un posible ataque o un patrón de tráfico anómalo.

- **Source IP:** Indica la dirección IP de origen del tráfico de red. Fundamental para identificar la fuente de las comunicaciones.
- **Destination IP:** Muestra la dirección IP de destino del tráfico de red para un flujo específico. Permite identificar el objetivo de las comunicaciones, lo cual es importante para determinar qué sistema están siendo afectado o atacados.
- **Flow Duration:** Registra la duración total, en segundos, de un flujo de tráfico de red desde su inicio hasta la finalización. Los patrones de duración de los flujos pueden ser indicativos de ataques.
- **Predicted_Type:** Es la clase de tráfico predicha por el modelo de detección de ataques DoS. Las clases pueden incluir tráfico benigno (BENIGN) o diferentes tipos de ataques DoS (MSSQL, UDP, Syn, UDPLag, NetBIOS, LDAP, Portmap. Representa la salida principal del clasificador.
- **Probabilidad de ataque:** Muestra la confianza del modelo en su predicción, expresada como un porcentaje, es la probabilidad calculada por el modelo de que el flujo de tráfico pertenezca a la clase predicha. Permite a los analistas evaluar la fiabilidad de la predicción del modelo, como una predicción de ataque indica una fuerte confianza en esa detección.
- **Es Ataque:** La última columna de la tabla indica el resultado de la clasificación del tráfico, diferenciando entre conexión normal y ataques. Para el tráfico legítimo, se muestra la etiqueta “Normal”, mientras que para las detecciones de amenazas se utiliza la etiqueta “Ataque”.

Ilustración 37

Tabla de Datos



3.4 Entorno de pruebas

Para la validación y demostración del sistema web de detección temprana de ataques DoS, se estableció un entorno de pruebas controlado. Lo que permitió simular escenarios de tráfico de red normal y la ejecución de ataques DoS.

A continuación, se presenta el entorno de pruebas:

3.4.1 Configuración de Máquinas Virtuales

Para establecer un entorno de pruebas controlado y aislado, se empleó la virtualización. Lo que es fundamental para la simulación de escenarios de red y realizar pruebas sin afectar la infraestructura de red productiva. Las máquinas virtuales tuvieron una configuración en modo puente (Bridge), lo que es crucial para que las máquinas puedan operar independientemente dentro de la misma red.

En las Ilustraciones 38, 39, 40, se detallan la configuración, lo que se muestra la especificación la capacidad de la máquina virtual. También se muestra el proceso de instalación y configuración de Kali Linux, incluyendo la asignación de nombres y credenciales.

- La Ilustración 41 demuestra la conectividad del equipo host hacia la máquina virtual atacante, que confirma una comunicación exitosa.
- Las Ilustraciones 42 y 43 muestran una exitosa prueba de conectividad realizada. Desde las máquinas virtuales Kali Linux.

Ilustración 38

Configuración VM

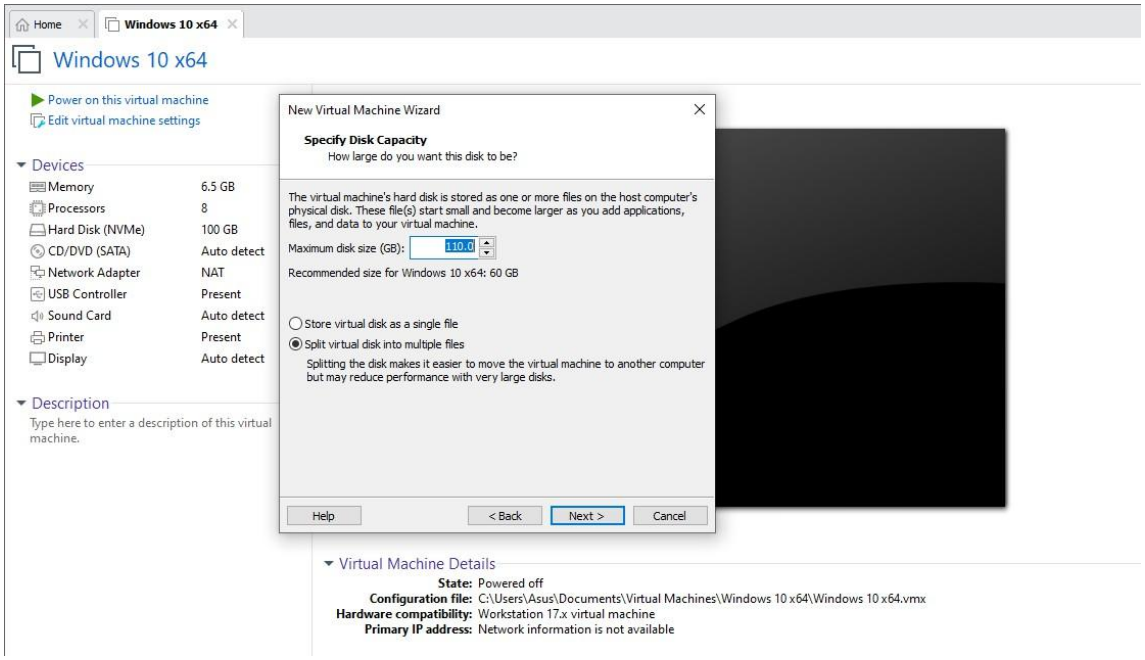


Ilustración 39

Configuración VM

Ilustración 40

Configuración VM

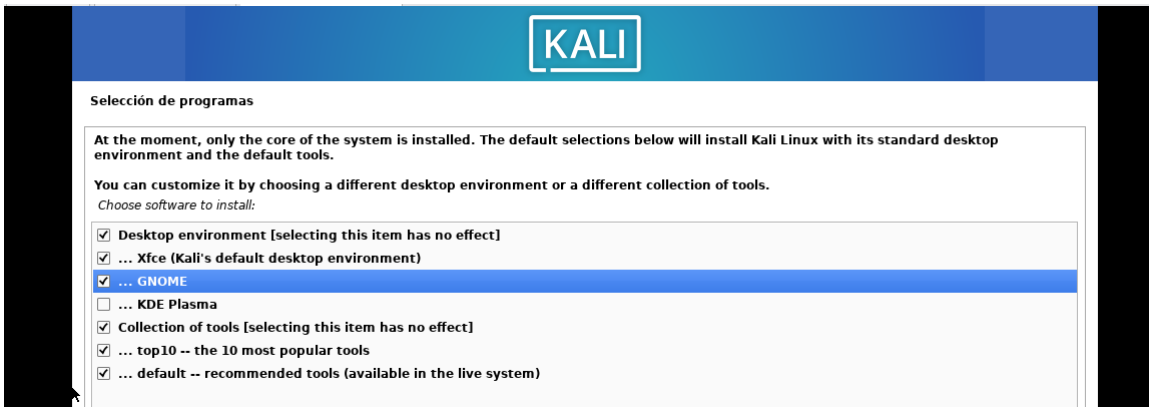


Ilustración 41

Ping de la máquina Host

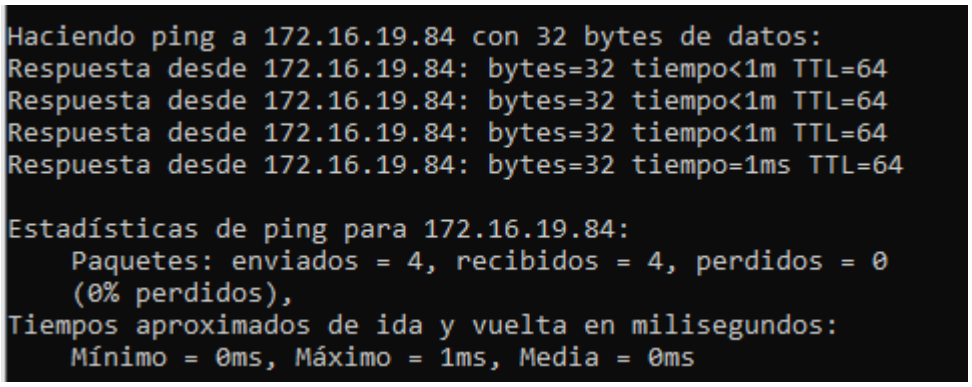


Ilustración 42

Ping de la máquina Atacante

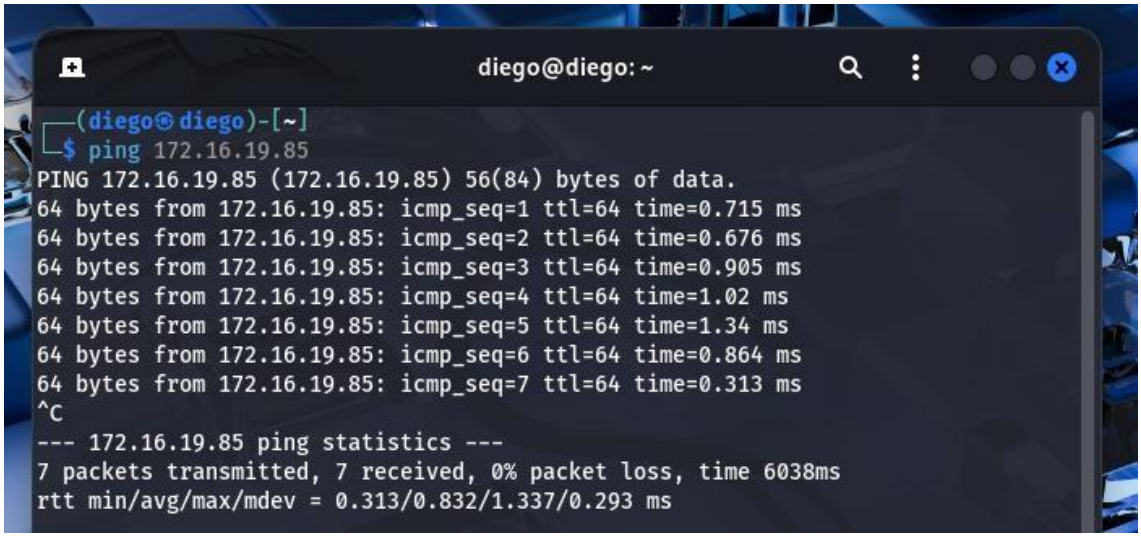
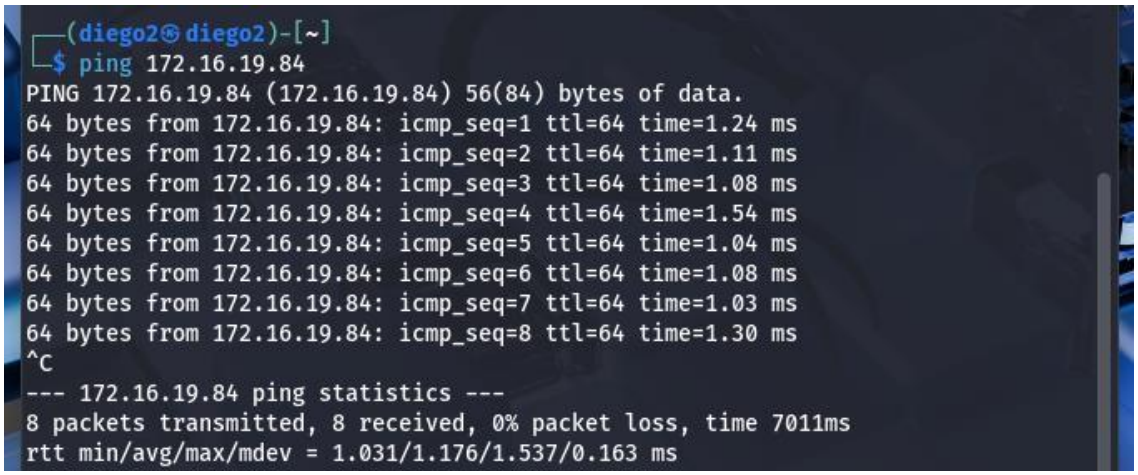


Ilustración 43

Ping de la máquina Recolección



3.4.2 Simulación y Recolección de Tráfico:

Para esta simulación de ataques DoS y la recolección se configuraron dos máquinas virtuales basadas en Kali Linux, una distribución especializada en pruebas de penetración y seguridad ofensiva.

- **Máquina Virtual Atacante:** En la Ilustración 44 se presenta el código del script en Python (syn_flood_scapy.py), desarrollado con la biblioteca Scapy para generar ataques SYN Flood, fue configurado para dirigir el ataque a una TARGET_IP: “172.16.19.80”, también se presenta el comando de ejecución para confirmar el ataque *sudo python3 syn_flood_scapy.py.*
- **Máquina Virtual Recolección de Tráfico:** En la Ilustración 56 se muestra el tráfico con la herramienta Wireshark para la captura del tráfico, donde se puede observar diversos protocolos y comunicaciones incluyendo a la víctima. Una vez finalizada la captura, en la Ilustración 57 se procede a guardar en formato. pcapng, posteriormente en la Ilustración 58 se revela el uso de tshark para poder procesar el archivo. pcapng y exportar columnas (frame. number, frame time, ip.src, ip.dst, ip.dst, _ws.col. Protocol, frame.len, _ws.col.Info) a un archivo según el nombre que se le haya puesto.

Ilustración 44

Ataque Syn_flood_scapy.py

```
 GNU nano 8.4                                    Búfer nuevo *
# /home/diego/syn_flood_scapy.py
from scapy.all import *
import random
import time

# --- CONFIGURACIÓN DEL ATAQUE ---
TARGET_IP = "172.16.19.80"  # <<<<<<<  (víctima)
TARGET_PORT = 80            # Puerto a atacar (ej. 80 para HTTP, 443 para HTTPS, 22 para SSH)

print(f"Iniciando ataque SYN Flood con Scapy a {TARGET_IP}:{TARGET_PORT} (origen SPOOFING aleatorio)")
print("Presiona Ctrl+C para detener el ataque.")

# --- Bucle de ataque ---
try:
    while True:
        src_ip = str(RandIP()) # Genera una IP de origen aleatoria (IP spoofing)
        src_port = random.randint(1024, 65535) # Puerto de origen aleatorio

        # Construir el paquete SYN
        packet = IP(src=src_ip, dst=TARGET_IP)/TCP(dport=TARGET_PORT, sport=src_port, flags="S")

        # Enviar el paquete (verbose=0 para no imprimir cada paquete en la terminal)
        send(packet, verbose=0)

        # Opcional: Pequeño retraso para controlar la velocidad
```

Ilustración 45

Captura de Tráfico

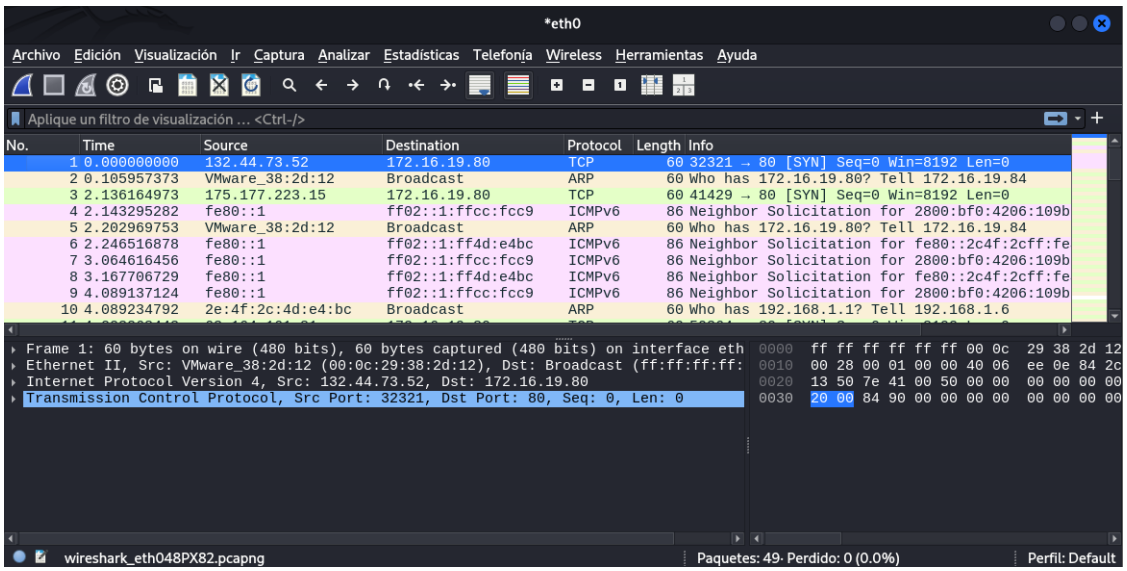


Ilustración 46

Formato pcapng

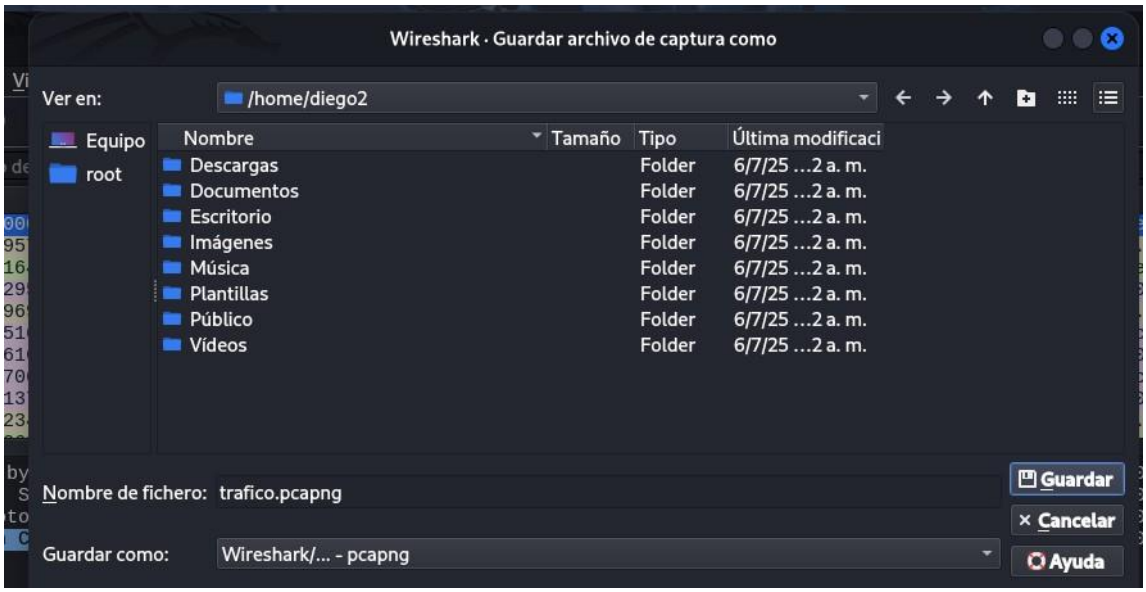


Ilustración 47

Extracción de Columnas

```
┌──(diego2㉿diego2)-[~]
└─$ tshark -r trafico.pcapng \
-T fields \
-e frame.number \
-e frame.time \
-e ip.src \
-e ip.dst \
-e _ws.col.Protocol \
-e frame.len \
-e _ws.col.Info \
-E header=y -E separator=, -E quote=d \
> trafico_basic_columns.csv
```

3.4.3 Interfaz de Usuario y Funcionalidades

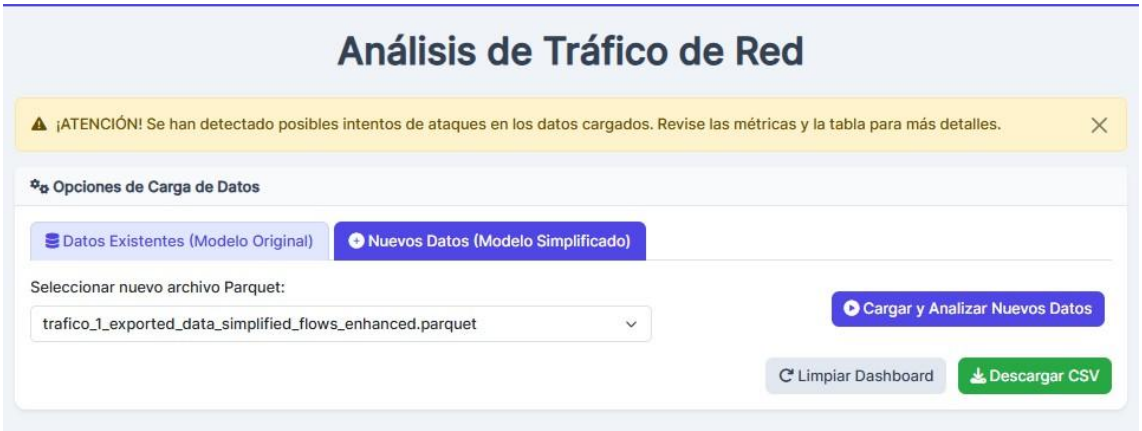
Una vez configurado el entorno de pruebas y los datos generados, capturados en formato csv, el sistema proporciona una interfaz intuitiva para la carga de datos, la visualización de rendimiento del modelo, el análisis gráfico de características y la exploración detallada de los datos de tráfico.

- En la ilustración 48 se muestra la pantalla, para procesar nuevos datos, el usuario debe seleccionar la pestaña “Nuevos Datos” lo que se procede a la carga del

archivo en este caso trafico_1_exported_data. parquet. Una vez seleccionado, se activará el botón “Cargar y Analizar Nuevos Datos” para iniciar el procesamiento.

Ilustración 48

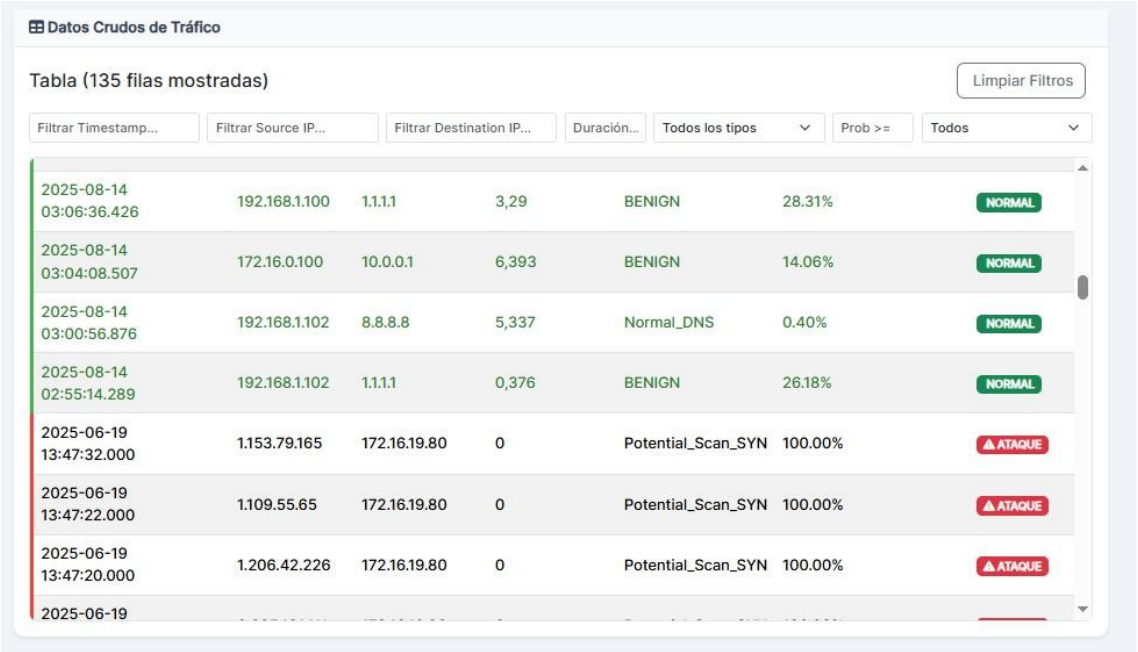
Selección y Carga de Archivos Nuevos Datos



- En Ilustración 49 se presenta la pestaña “Tabla de Datos”, se identifican claramente los patrones que corresponden a tráfico normal, que presentan duraciones y bajas probabilidades de ataque, el otro patrón muestra eventos sospechosos clasificados como posibles escaneos SYN, con probabilidad de ataque, caracterizados por flujos de duración que reflejan intentos de reconocimientos de puertos y servicios, evidenciando así una marcada distinción entre tráfico normal y actividad maliciosa relacionada con los escaneos en la red.

Ilustración 49

Tabla de Datos



3.4.4 Pestaña de Monitoreo

Como complemento a las funcionalidades de análisis de datos ya existentes, se implementó una nueva pestaña nombrada como “Monitoreo”, que permite la visualización y el seguimiento continuo de tráfico de red capturado por el sistema. Esta funcionalidad representa una extensión significativa para el sistema, al ofrecer una interfaz dedicada para el monitoreo de conexiones y la identificación de amenazas.

Características principales del módulo de monitoreo

La pestaña de monitoreo se integra de forma fluida con el sistema actual, manteniendo la coherencia visual de la interfaz. Como se observa en la Ilustración 50.

Ilustración 50

Pestaña Monitoreo

Interfaz de Monitoreo de Conexiones

Al seleccionar la pestaña monitore, el sistema despliega una interfaz específicamente diseñada para la visualización de conexiones de red activas. La interfaz, se observa en las figuras 51 y 52, que presenta lo siguiente:

- **Panel de Control:** Ubicado en la parte superior incluye botones de “Iniciar” y “Detener” para el control del proceso.
- **Tabla de Conexiones:** Visualización tabular detallada de las conexiones.

Ilustración 51

Vista Principal

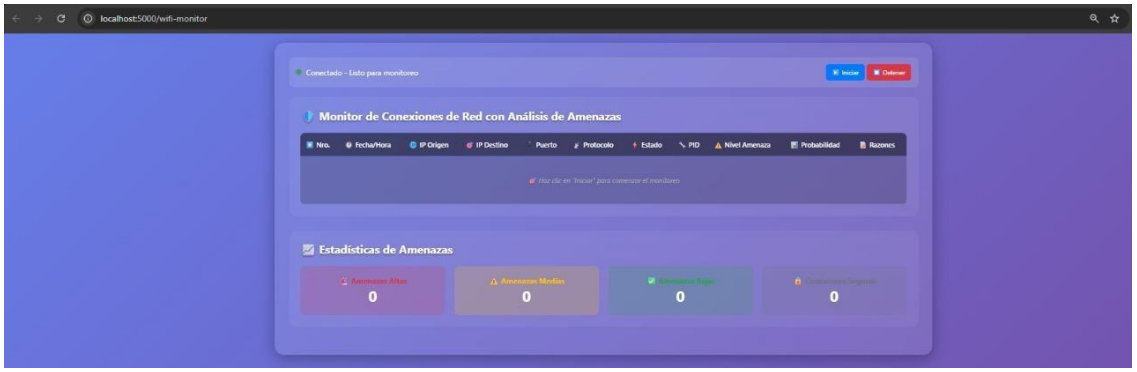
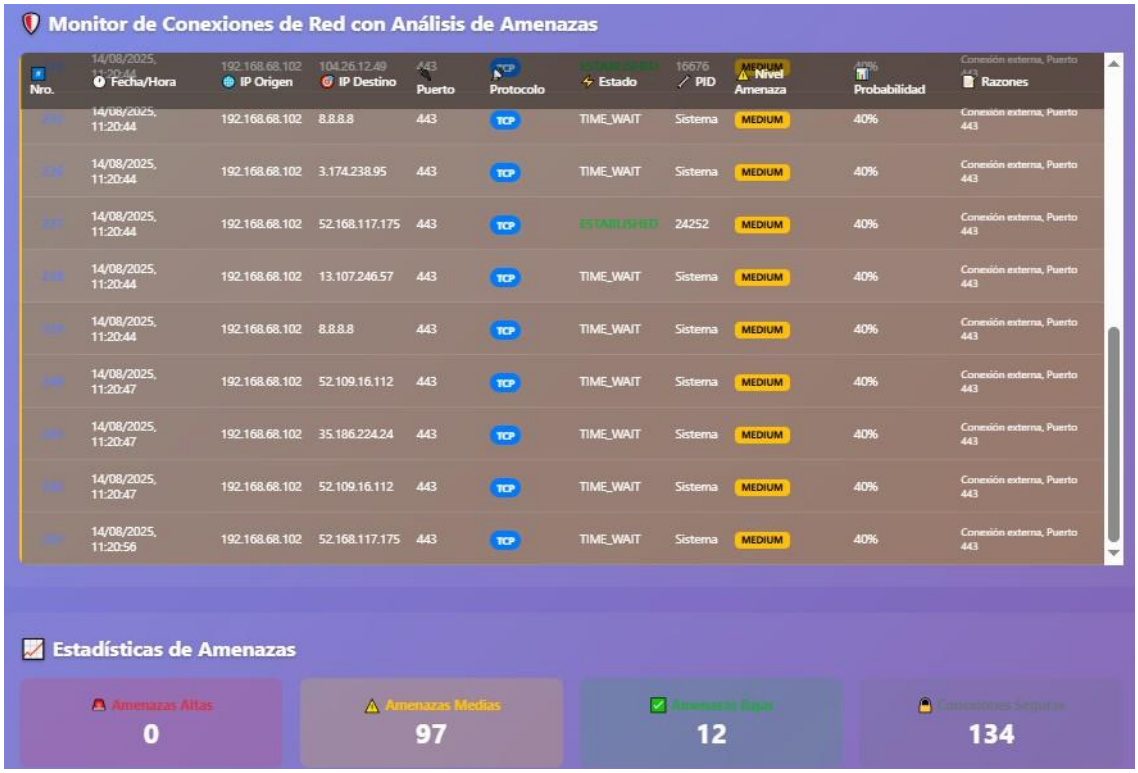


Ilustración 52

Detalle de conexión



Análisis y Clasificación de Amenazas

El sistema incorpora un mecanismo automático que evalúa cada conexión capturada, asignándose un nivel de amenaza basado en patrones de comportamiento y características específicas:

- **Nro.:** Número secuencial que identifica cada conexión registrada.
- **Fecha/Hora:** Marca precisa que indica cuándo se estableció o detectó la conexión.
- **IP Origen:** Dirección IP del equipo que inicia la conexión.
- **IP Destino:** Dirección IP del equipo receptor de la conexión.
- **Puerto:** Puerto de red utilizado para la comunicación.
- **Protocolo:** Protocolo de red.
- **Estado:** Estado de la conexión.
- **PID:** Identificador del proceso asociado a la conexión.
- **Probabilidad:** Porcentaje que indica la certeza en la clasificación del nivel de amenaza.
- **Razones:** Descripción detallada de los motivos que justifican clasificación.

Análisis y Clasificación de Amenazas

El sistema incorpora un mecanismo automático que evalúa cada conexión capturada, asignándole un nivel de amenaza basado en patrones de comportamiento y características.

- Las conexiones normales se clasifican como “NONE”, con bajos niveles de probabilidad.
- Las conexiones sospechosas se identifican como “MEDIUM”, con probabilidades moderadas.
- Las conexiones de alto riesgo son clasificadas según el tipo especifico detectada.

Por ejemplo, en la figura 71 se muestra un caso particular donde el sistema detectó una conexión externa con nivel “MEDIUM”, etiquetada “Conexión externa, Puerto 443”.

Estadística de Amenazas

En la parte inferior de la interfaz, se presenta un panel con estadísticas que resumen cuantitativamente las amenazas detectadas:

- **Amenazas Altas:** Conteo de amenazas consideradas de alto riesgo.
- **Amenazas Medias:** Número de amenazas clasificadas como riesgo moderad.
- **Amenazas Bajas:** Cantidad de amenazas de bajo riesgo.
- **Total de Conexiones:** Número de conexiones monitoreadas.

Captura y Procesamiento de Tráfico

El módulo de monitoreo opera mediante la captura continua de paquetes, de red, procesado cada conexión con un modelo de machine learning previamente entrenado. El sistema analiza características del flujo, patrones de comunicación y otros indicadores relevantes para determinar la probabilidad de que una conexión represente una amenaza de seguridad.

Para una mejor usabilidad, que el sistema incorpora alertas visuales mediante colores basado en el nivel de amenaza, posibilitando la identificación rápido de situaciones que requieren atención.

CONCLUSIONES

Alta Capacidad de Detección el RandomForestClassifier, configurado con class_weight='balanced' y entrenado con características de flujo de tráfico (fwd_pkt_len_max, min_pkt_len, fwd_seg_size_avg), demostró una alta precisión y exhaustividad en la identificación temprana de diversos ataques DoS (MSSQL, UDP, Syn, etc.), incluso en datasets desequilibrados.

Sistema Web Funcional e Intuitivo fue la implementación de una arquitectura web con Flask (Python) y Bootstrap (JavaScript) facilitó la interacción del usuario para la carga de datos, la visualización detallada del tráfico y la presentación clara de los resultados del análisis y las métricas de rendimiento del modelo. Esto permite una monitorización activa y la detección de anomalías.

Manejo Eficiente de Grandes Dato la estrategia del procesamiento en lotes con StandardScaler (partial_fit) y el muestreo estratificado de datos Parquet (MAX_TRAINING_SAMPLES = 1,000,000) optimizaron el uso de recursos, permitiendo el entrenamiento del modelo con grandes volúmenes de información de tráfico de red, lo cual es crucial para la aplicabilidad en entornos reales.

RECOMENDACIONES

Expansión de la Base de Conocimiento de Ataques poder integrar nuevos datasets que incluyan una mayor variedad de ataques DoS (ej., DDoS, ataques a la capa de aplicación) y tráfico malicioso emergente, así como datos de redes Wi-Fi reales, para mejorar la generalización y adaptabilidad del modelo.

Optimización y Evaluación Continua del Modelo a investigar para la aplicación de técnicas avanzadas de ajuste de hiperparámetros (Hyperparameter Tuning) y explorar otros algoritmos de Machine Learning o enfoques de Deep Learning para evaluar potenciales mejoras en la precisión y eficiencia de detección.

BIBLIOGRAFIA

(CIC), C. I. (julio de 2019). *Canadian Institute for Cybersecurity – University of New Brunswick.* Obtenido de chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://oa.upm.es/79025/1/TFG_DAVID_RAMOS_ARCHILLA.pdf

ACT Fibernet. (2024). *ACT Fibernet Blog.* Obtenido de ACT Fibernet Blog: https://www.actcorp.in/blog/common-challenges-and-limitations-wifi-networks

Almadhor, A. A. (2024). *Nature*. Obtenido de https://www.nature.com/articles/s41598-024-76016-6

Apache Software Foundation. (2024). *Apache Flink ML Documentation.* Obtenido de https://nightlies.apache.org/flink/flink-ml-docs-release-2.1/docs/operators/classification/linearsvc/#:~:text=Linear%20Support%20Vector%20Machine%20(Linear,the%20distance%20between%20classified%20samples.

Arriola, J., & Gómez, D. (2024). *MDPI – Technologies.* Obtenido de MDPI – Technologies: https://www.mdpi.com/2227-7080/13/3/88

Authors, B. (2023). *getbootstrap.com .* Obtenido de https://getbootstrap.com/

Beck, K. y. (2023). *Asana.* Obtenido de Asana: https://asana.com/es/resources/extreme-programming-xp

Berríos, S., Garcia, S., Hermosilla, P., & Allende-Cid, H. (2025). *A Machine-Learning-Based Approach for the Detection and Mitigation of Distributed Denial-of-Service Attacks in Internet of Things Environments.* Obtenido de https://www.mdpi.com/2076-3417/15/11/6012?

Chakray. (s.f.). *¿CUÁLES SON LAS VENTAJAS DE UNA API REST?* Obtenido de Chakray: https://www.chakray.com/es/cuales-son-las-ventajas-de-una-api-rest/#:~:text=Las%20razones%20por%20las%20que,que%20ello%20represente%20muchas%20dificultades.

Cloudflare. (2024). *Cloudflare.* Obtenido de Cloudflare.: https://www.cloudflare.com/es-es/learning/ddos/glossary/denial-of-service/

Cloudflare. (s.f.). *Cloudflare.* Obtenido de Cloudflare: https://www.cloudflare.com/es-es/learning/ddos/glossary/denial-of-service/

contributors, W. (2025). *Wikipedia.* Obtenido de https://en.wikipedia.org/wiki/Isolation_forest

Cuvi, B. (Marzo de 2023). *Análisis de la vulnerabilidad del sistema de conexión a la red WI-FI genérica de un Instituto Educativo*. Obtenido de https://repositorio.puce.edu.ec/server/api/core/bitstreams/ba205440-8282-425a-8158-f1aa4ca7142a/content

D. Narváez, C. R. (Enero de 2020). *Evaluacion de ataques de Denegación de servicio DoS y DDoS, y mecanismos de protección.* Obtenido de https://journal.espe.edu.ec/ojs/index.php/geeks/article/view/249/226

D. Narváez, C. R. (Enero de 2020). *Evaluación de ataques de Denegación de servicio DoS y DDoS, y mecanismos de protección.* Obtenido de Evaluación de ataques de Denegación de servicio DoS y DDoS, y mecanismos de protección: https://journal.espe.edu.ec/ojs/index.php/geeks/article/view/249/226

Developers, J. (2023). *joblib.readthedocs.io.* Obtenido de https://joblib.readthedocs.io/en/stable/

Europea, U. (2021). *Aprendizaje supervisado y no supervisado.* Obtenido de Aprendizaje supervisado y no supervisado: https://universidadeuropea.com/blog/aprendizaje-supervisado-no-supervisado/

Experts®, World Wide WiFi. (2020). *World Wide WiFi Experts®.* Obtenido de World Wide WiFi Experts®: https://worldwide-wifi-experts.com/services/design/architecture

Flask-CORS Developers. (2024). *Flask-CORS Documentation.* Obtenido de https://flask-cors.readthedocs.io/en/latest/

Fortinet. (s.f.). *Fortinet CyberGlossary.* Obtenido de Fortinet CyberGlossary: https://www.fortinet.com/lat/resources/cyberglossary/types-of-cyber-attacks

Foundation, A. S. (2024). *Apache Arrow Documentation.* Obtenido de https://arrow.apache.org/docs/python/parquet.html

Foundation, P. S. (2024). *Python.* Obtenido de https://docs.python.org/es/3/tutorial/

Foundation, P. S. (2024). *Python Docs.* Obtenido de https://docs.python.org/3/library/

Foundation, P. S. (2024). *Python Documentation.* Obtenido de https://docs.python.org/3/library/logging.html

Foundation, P. S. (2024). *Python Documentation.* Obtenido de https://docs.python.org/3/library/sys.html

Foundation, T. W. (2024). *Wireshark.* Obtenido de https://www.wireshark.org/

Grinberg, M. (2022). *flask.palletsprojects.com.* Obtenido de https://flask.palletsprojects.com/en/stable/

Hat, R. (27 de Julio de 2020). *¿Qué es la arquitectura orientada a los servicios (SOA)?* Obtenido de Red Hat: https://www.redhat.com/es/topics/cloud-native-apps/what-is-service-oriented-architecture

Hat, R. (27 de Julio de 2020). *Red Hat.* Obtenido de Red Hat: https://www.redhat.com/es/topics/cloud-native-apps/what-is-service-oriented-architecture

Hat, R. (27 de Julio de 2020). *Red Hat*. Obtenido de Red Hat: https://www.redhat.com/es/topics/cloud-native-apps/what-is-service-oriented-architecture

Hunter, J. D. (2023). *matplotlib.org.* Obtenido de matplotlib.org

Hunter, J. D. (2023). *matplotlib.org .* Obtenido de https://matplotlib.org/stable/

Ian Goodfellow, Y. B. (2016). Deep Learning. 98-164. Obtenido de http://alvarestech.com/temp/deep/Deep%20Learning%20by%20Ian%20Goodfellow,%20Yoshua%20Bengio,%20Aaron%20Courville%20(z-lib.org).pdf

IBM Developer Team. (2023). *IBM Developer.* Obtenido de https://developer.ibm.com/tutorials/awb-implement-xgboost-in-python/

imbalanced-learn Developers. (2024). *imbalanced-learn Documentation.* Obtenido de https://imbalanced-learn.org/stable/

imbalanced-learn Developers. (2024). *imbalanced-learn Documentation.* Obtenido de https://imbalanced-learn.org/stable/references/generated/imblearn.ensemble.BalancedRandomForestClassifier.html

James F. Kurose, K. W. (2017). *Universidad de Montenegro.* Pearson .

Johnson, A. &. (2021). Machine Learning Applications in Network Security. *IEEE Transactions on Cybersecurity*, 112-129.

Jurado, F., Escobar, E., & Carrión, J. (2021). La seguridad de la información de las microempresas en el Ecuador. *Revista Científica ECOCIENCIA*, 48–64.

Kaggle. (2017). *Kaggle.* Obtenido de https://www.kaggle.com/code/clemensmzr/weighted-random-forest

Maciej Bartłomiej Sikora. (2024). *NordPass.* Obtenido de NordPass: https://nordpass.com/es/blog/machine-learning-in-cybersecurity/

ManageEngine. (2025). *¿Qué es un ataque de denegación de servicio?* Obtenido de ManageEngine NetFlow Analyzer: https://www.manageengine.com/latam/netflow/que-es-el-ataque-de-denegacion-de-servicios.html

Manapro Consultores. (s.f.). *Manapro*. Obtenido de https://manapro.com/25-ha-reportado-ciberataques-pymes-latinoamericanas-en-riesgo/

Mero, E., Ortiz, M., Yanangómez, J., & Rodríguez, A. (2021). Análisis de seguridad de las redes inalámbricas en las redes inalámbricas en la carrteta de Tecnologìas de la Información. *Serie Científica de la Universidad de las Ciencias Informáticas*, 139-147.

Microsoft. (2023). *Microsoft Support.* Obtenido de Microsoft Support: https://support.microsoft.com/es-es/topic/conceptos-b%C3%A1sicos-sobre-bases-de-datos-a849ac16-07c7-4a31-9948-3c8c94a7c204

Morales Tapia, J. B. (2023). *Repositorio UTMachala.* Obtenido de Sistema de detección de ataques de denegación de servicios basado en aprendizaje automático: https://repositorio.utmachala.edu.ec/handle/48000/23572

Networks, J. (2023). *Juniper.* Obtenido de Juniper: https://www.juniper.net/mx/es/research-topics/what-is-an-access-point-in-networking.html

nibusinessinfo.co.uk. (s.f.). *nibusinessinfo.co.uk.* Obtenido de nibusinessinfo.co.uk: https://www.nibusinessinfo.co.uk/content/security-issues-wireless-networks#:~:text=Wireless%20network%20security%20issues%20and%20threats&text

=spoofing%20and%20session%20hijacking%20%2D%20where,transmitted%20over%20the%20secure%20network

Nick Hotz. (2024). *Data Science PM.* Obtenido de https://www.datascience-pm.com/kdd-and-data-mining/

Noguera, D. (2023). *FlashStart.* Obtenido de FlashStart: https://flashstart.com/es/son-seguras-las-redes-wifi-descubre-los-riesgos-y-vulnerabilidades/

Oracle. (2023). *Oracle.* Obtenido de Oracle: https://www.oracle.com/mx/database/what-is-database/

Osundare, O. S., Ike, C. S., Fakeyede, O. G., & Ige, A. B. (2022). *Engineering Science & Technology Journal.* Obtenido de Engineering Science & Technology Journal.

Ouhssini, Karim Afdel, Mohamed Akouhar, Elhafed Agherrabi, Abdallah Abarda. (2024). *ResearchGate.* Obtenido de https://www.researchgate.net/publication/383457082_Advancements_in_detecting_preventing_and_mitigating_DDoS_attacks_in_cloud_environments_A_comprehensive_systematic_review_of_state-of-the-art_approaches

Pedregosa, F. V. (2023). *scikit-learn.* Obtenido de https://scikit-learn.org/stable/

Peral, A. V. (2019). *Repositorio Institucional (O2) de la Universitat Oberta de Catalunya (UOC)*. Obtenido de Repositorio Institucional (O2) de la Universitat Oberta de Catalunya (UOC): https://openaccess.uoc.edu/bitstream/10609/97586/8/avalenciapTFM0619memoria.pdf

Proofpoint, I. (2024). *Proofpoint.* Obtenido de Proofpoint: https://www.proofpoint.com/es/threat-reference/wifi#:~:text=Las%20redes%20wifi%20funcionan%20mediante,5%20GHz%20y%206%20GHz.

Radware. (s.f.). *Radware CyberPedia.* Obtenido de Radware CyberPedia: https://es.radware.com/cyberpedia/ddos-attacks/dos-vs-ddos-attack-what-is-the-difference/

Rodriguez, d. l. (2021). Aplicaciones educativas digitales y la falta de seguridad de los datos personales de sus usuarios. *RIDE Revista Iberoamericana Para La Investigación Y El Desarrollo Educativo.*

Romero, M., & Jumbo, R. (2019). ANÁLISIS DE VULNERABILIDADES DE REDES INALÁMBRICAS PARA EVITAR LA INSEGURIDAD DE LA INFORMACIÓN DE LOS USUARIOS EN EL LABORATORIO DE TELECOMUNICACIONES DE LA CARRERA DE INGENIERÍA EN COMPUTACIÓN Y REDES. *Tesis.* Universidad Estatal de Manabi, Manabi. Obtenido de http://repositorio.unesum.edu.ec/handle/53000/1952

Russo, C., Ramón, H., Alonso, N., Cicerchia, B., Esnaola, L., & Tessore, J. P. (s.f.). *Tratamiento Masivo de Datos Utilizando Técnicas de Machine Learning.* Junín, Argentina: Universidad Nacional del Noroeste de la Provincia de Buenos Aires (UNNOBA).

S. Abiramasundari y V. Ramaswamy. (2025). *Nature*. Obtenido de https://www.nature.com/articles/s41598-024-84879-y

S.M.K. (2018). *Wireless Network Security: A Comprehensive Guide to Security Protocols and Solutions.* Obtenido de Wireless Network Security: A Comprehensive Guide to Security Protocols and Solutions: https://bevijaygupta.medium.com/a-comprehensive-guide-to-wireless-network-security-37d18aeabbce

Sadiq, M. &. (2018). *ResearchGate.* Obtenido de ResearchGate: https://www.researchgate.net/publication/330569376_The_Role_of_Artificial_Intelligence_in_Cyber_Security

Salazar,Barahona,Delgado,Suaréz, A. (2023). Seguridad en Redes Wifi. *Tesis de Maestría.* UIDE, Quito. Obtenido de https://repositorio.uide.edu.ec/bitstream/37000/6106/1/UIDE-Q-TMCSE-2022-13.pdf

Scikit-learn Developers. (2024). *Scikit-learn Documentation.* Obtenido de https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html

Scikit-learn Developers. (2024). *Scikit-learn Documentation.* Obtenido de Scikit-learn Documentation: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

Seaborn Developers. (2024). *Seaborn Documentation.* Obtenido de https://seaborn.pydata.org/tutorial/introduction.html

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2019). *ResearchGate.* Obtenido de https://www.researchgate.net/publication/336953914_Developing_Realistic_Distributed_Denial_of_Service_DDoS_Attack_Dataset_and_Taxonomy

Sharma, P. &. (2023). *ScienceDirect*. Obtenido de https://www.sciencedirect.com/science/article/abs/pii/S1570870523002421

Sikora, M. B. (2024). *NordPass.* Obtenido de NordPass: https://nordpass.com/es/blog/machine-learning-in-cybersecurity/

Soafumc. (21 de Agosto de 2016). *VENTAJAS Y DESVENTAJAS DEL SOA*. Obtenido de VENTAJAS Y DESVENTAJAS DEL SOA: https://yamilpo.wordpress.com/2016/08/21/ventajas-y-desventajas-del-soa/

Software, A. (2021). *Avast.* Obtenido de Avast: https://www.avast.com/es-es/c-wep-vs-wpa-or-wpa2

Solórzano, Rodriguez, Anzules,Mar, W. (2022). Redes inalámbricas, su incidencia en la privacidad de la información. *Tema de Tesis.* Universidad Estatal de Manabi, Manabi. Obtenido de https://revistas.unesum.edu.ec/JTI/index.php/JTI/article/view/25/42

Sreeram y Saravanan. (2020). *MDPI – Electronics.* Obtenido de https://www.mdpi.com/2079-9292/9/10/1689?utm_source=chatgpt.com

Staff, T. E. (2024). *SearchSecurity.* Obtenido de https://www.techtarget.com/searchsecurity/definition/Wired-Equivalent-Privacy

Stallings, W. (2014). *elhacker.info.* Obtenido de elhacker.info: https://elhacker.info/manuales/Redes/3._Network-security-essentials-4th-edition-william-stallings.pdf

Team), E. (. (2025). *ArcGIS Pro GeoAI Documentation.* Obtenido de https://pro.arcgis.com/en/pro-app/latest/tool-reference/geoai/how-lightgbm-works.htm#:~:text=LightGBM%20is%20a%20gradient%20boosting,high%20performance%20with%20distributed%20systems.

Team, C. (2023). *Naukri Code360.* Obtenido de https://www.naukri.com/code360/library/shutil-module-in-python

Team, D. D. (2023). *dask.org.* Obtenido de https://www.dask.org/

team, T. p. (2023). *pandas.pydata.org.* Obtenido de https://pandas.pydata.org/

Universidad Internacional de Valencia. (21 de Marzo de 2018). *Equipo de Expertos en Ciencia y Tecnología.* Obtenido de Equipo de Expertos en Ciencia y Tecnología: https://www.universidadviu.com/int/actualidad/nuestros-expertos/que-es-la-seguridad-informatica-y-como-puede-ayudarme

Vacca, J. R. (2017). *Computer and Information Security Handbook.* Cambridge, MA: Morgan Kaufmann. Obtenido de Elsevier: https://booksite.elsevier.com/samplechapters/9780123704719/Sample_Chapters/01~Front_Matter.pdf

Valencia, U. I. (2024). *Universidad VIU.* Obtenido de Universidad VIU: https://www.universidadviu.com/ec/maestria-ciberseguridad?var=no&c=I90503M7002&gad_source=1&gclid=Cj0KCQiAsaS7BhDPARIsAAX5cSA7_pO9o3-CRmgzRuvJmiZoLU1dtjvimhQkXs2d7_UOVnjjZrzq2YwaAitSEALw_wcB&gclsrc=aw.ds

Wikipedia. (2024). *Wikipedia.* Obtenido de https://es.wikipedia.org/wiki/Isolation_forest

Wikipedia contributors. (2025). *Máquina de vectores de soporte*. Obtenido de Wikipedia, la enciclopedia libre: https://es.wikipedia.org/wiki/M%C3%A1quina_de_vectores_de_soporte

Wikipedia contributors. (2025). *Random forest*. Obtenido de Wikipedia, la enciclopedia libre: https://es.wikipedia.org/wiki/Random_forest?utm_source=chatgpt.com

Wikipedia contributors. (2025). *Red neuronal artificial*. Obtenido de Wikipedia, la enciclopedia libre: https://es.wikipedia.org/wiki/Red_neuronal_artificial?utm_source=chatgpt.com

ANEXOS

Pontificia Universidad Católica del Ecuador
Seréis mis testigos | IBARRA

OFICINA DE TECNOLOGÍAS INFORMÁTICAS

CARTA DE ACEPTACIÓN

Ibarra, 12 de agosto de 2025

Msc. José Tamayo
DIRECTOR DE LA ESCUELA DE HÁBITAT, INGENIO Y CREATIVIDAD DE LA PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE- IBARRA
Presente.

Estimado Diego Carlos Pabón Ramírez,

En nombre de la Oficina de Tecnologías Informáticas de la PUCE IBARRA, me complace informar que hemos recibido y revisado el proyecto de titulación: "SISTEMA WEB BASADO EN APRENDIZAJE AUTOMÁTICO PARA LA DETECCIÓN TEMPRANA DE ATAQUES DE DENEGACIÓN DE SERVICIO (DOS) EN REDES WIFI" que ha desarrollado en nuestra unidad.
Queremos expresar nuestra satisfacción con el trabajo realizado, demostrando ser una contribución importante para nuestra área de redes, proporcionando una solución para la detección temprana de ataques DoS en redes WiFi, la implementación de técnicas de aprendizaje automático para nuestra capacidad de monitoreo y proyección de redes inalámbricas.

Agradecemos su dedicación y profesionalismo demostrado a lo largo del desarrollo de este proyecto. Estamos seguros de que los conocimientos y la experiencia que ha adquirido serán de gran valor para su carrera profesional.

Atentamente,

Firmado digitalmente por fsanchez@puces i.edu.ec

MSc. Franklin Sánchez E.
Coordinador de la Oficina de Tecnologías Informáticas

Dirección: Av. Jorge Guzmán Rueda y Av. Aurelio Espinosa Pólit. Ciudadela "La Victoria".
Teléf: (593-6) 2615 500 / 2615 453 **Ext.** 1000 **Cel.** 099 236 27 13 / 098 138 3498
Ibarra - Ecuador / www.pucesi.edu.ec