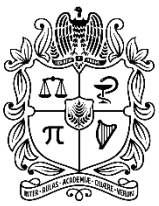


UNIVERSIDAD
NACIONAL
DE COLOMBIA

Implementación de un Sistema de Detección de Intrusos Soportado en Técnicas de Aprendizaje Supervisado Orientado a Servicios en la Nube para la Detección de Ataques de Denegación de Servicios Distribuidos

José Albeiro Montes Gil

Universidad Nacional de Colombia
Facultad de Administración Manizales, Colombia
Manizales, Colombia
2023

Implementación de un Sistema de Detección de Intrusos Soportado en Técnicas de Aprendizaje Supervisado Orientado a Servicios en la Nube para la Detección de Ataques de Denegación de Servicios Distribuidos

José Albeiro Montes Gil

Tesis presentada como requisito parcial para optar al título de:

Magister en Administración de Sistemas Informáticos

Director:

Ph.D. Néstor Darío Duque Méndez

Codirector:

Ph.D. Gustavo Adolfo Isaza Echeverri

Línea de Investigación:

Inteligencia Artificial

Grupo de Investigación:

GAIA – Grupo de Ambientes Inteligentes Adaptativos

Universidad Nacional de Colombia

Facultad de Administración, Departamento de Administración

Manizales, Colombia

2023

Dedicatoria

A mis padres, por su amor y apoyo incondicional. Todo lo que soy, ha sido por ustedes, los amo. A mi pequeño hermano, por llenar mi vida de felicidad, a mi hermana por siempre creer en mí, a Lizeth Mateus Herrera por su amor, y a Carlos Alberto Valencia Salcedo, siempre en mi corazón.

Declaración de obra original

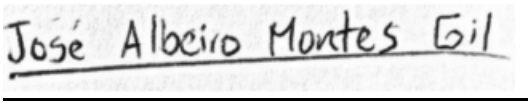
Yo declaro lo siguiente:

He leído el Acuerdo 035 de 2003 del Consejo Académico de la Universidad Nacional. «Reglamento sobre propiedad intelectual» y la Normatividad Nacional relacionada al respeto de los derechos de autor. Esta disertación representa mi trabajo original, excepto donde he reconocido las ideas, las palabras, o materiales de otros autores.

Cuando se han presentado ideas o palabras de otros autores en esta disertación, he realizado su respectivo reconocimiento aplicando correctamente los esquemas de citas y referencias bibliográficas en el estilo requerido.

He obtenido el permiso del autor o editor para incluir cualquier material con derechos de autor (por ejemplo, tablas, figuras, instrumentos de encuesta o grandes porciones de texto).

Por último, he sometido esta disertación a la herramienta de integridad académica, definida por la universidad.



Nombre

Fecha 31/01/2023

Agradecimientos

A mis padres, por su apoyo incondicional.

A mi director el profesor Néstor Darío Duque Méndez por su apoyo en todo mi proceso de formación desde hace varios años. A mi codirector el profesor Gustavo Adolfo Isaza Echeverri por su colaboración en todo el proceso de investigación.

A la profesora Valentina Tabares Morales por toda su colaboración dentro y fuera del marco de esta investigación. Al profesor Jefferson Arango López por su disposición de apoyo desde el primer momento. A la profesora Ana Lorena Uribe Hurtado por su tiempo y aportes en este proceso. A Fabián Alberto Ramírez Rodríguez por su colaboración en la investigación. A mis amigos, Santiago Torres Jaramillo, Santiago Parra Giraldo, Sergio Aguirre Serrano, Santiago Aguirre Gálvez, Juan David Sánchez, Diego David Ramos, Santiago Montoya Salazar, Johan Gilberto Garcés y Cristian Camilo García.

A mis compañeros, Luis Felipe García, Aldemir Vargas Eudor, Paula Taborda, Laura Valentina Carreño, Daniela Escobar, María José Giraldo y Julián Pachón por su apoyo durante las pruebas realizadas.
Al Grupo de Ambientes Inteligentes Adaptativos (GAIA) por abrirme las puertas en el mundo académico desde el año 2017.

A todos aquellos que contribuyeron de diferentes formas en el cumplimiento de esta meta.

Implementación de un Sistema de Detección de Intrusos Soportado en Técnicas de Aprendizaje Supervisado Orientado a Servicios en la Nube para la Detección de Ataques de Denegación de Servicios Distribuidos

Resumen

Dados los avances presentados en la actualidad en el área de las Tecnologías de la Información y las Comunicaciones, la dependencia de las organizaciones hacia los activos tecnológicos cada día es más importante, razón por la cual, el área de seguridad informática tiene la responsabilidad de proporcionar mecanismos que garanticen la protección de la infraestructura tecnológica. Sin embargo, actualmente son constantes los ataques informáticos, los cuales buscan afectar la disponibilidad, integridad o confidencialidad de los datos y la información. A pesar de los numerosos mecanismos de seguridad con los que se cuenta actualmente, los atacantes logran vulnerar los diferentes mecanismos de protección, en particular, realizando ataques de Denegación de Servicios (DoS) y Denegación de Servicios Distribuidos (DDoS). Teniendo en cuenta que a pesar de la implementación de sistemas de seguridad tradicionales, no se ha conseguido una mitigación de los ataques en su totalidad, la adaptación de técnicas de aprendizaje supervisado para la detección de ataques de tipo DoS/DDoS es viable, dada la capacidad de los algoritmos de inteligencia artificial para clasificar y emitir predicciones. La comunidad científica respalda ampliamente la propuesta de implementar Sistemas de Detección de Intrusos usando técnicas de inteligencia artificial, no obstante, las soluciones desarrolladas no están orientadas a usuarios administradores de seguridad en redes sin conocimientos en aprendizaje de máquina y con la generación de reportes dinámicos y con carácter estadístico orientado a servicios en la nube. En esta tesis de maestría, se propuso el diseño e implementación de una arquitectura orientada a servicios en la nube, la selección de las técnicas de aprendizaje supervisado más relevantes en la detección de ataques DoS/DDoS y la implementación del sistema de Detección de Intrusos. El prototipo demuestra que las técnicas de aprendizaje supervisado pueden ser implementadas como servicios en la nube, garantizando su desempeño en la detección de este tipo de ataques en redes físicas y en tiempo real.

Palabras clave: IDS, SOA, Attacks, DoS, DDoS, Aprendizaje de Máquina.

Implementation of an Intrusion Detection System Supported by Supervised Service-Oriented Learning Techniques in the Cloud for the Detection of Distributed Denial of Service Attacks.

Abstract

Given the advances presented today in the field of Information and Communication Technologies, the dependence of organizations on technological assets is becoming increasingly important. Therefore, the area of computer security has the responsibility to provide mechanisms that ensure the protection of technological infrastructure. However, cyberattacks seeking to affect the availability, integrity, or confidentiality of data and information are becoming increasingly constant. Despite the numerous security mechanisms currently available, attackers manage to compromise different protection mechanisms, particularly by carrying out Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks. Considering that traditional security systems have not achieved complete mitigation of attacks, the adaptation of supervised learning techniques for DoS/DDoS attack detection is viable given the ability of artificial intelligence algorithms to classify and make predictions. The scientific community widely supports the proposal to implement Intrusion Detection Systems using artificial intelligence techniques. However, the solutions developed are not aimed at security administrators in networks without knowledge of machine learning and with the generation of dynamic and statistical reports oriented towards cloud services. This master's thesis proposes the design and implementation of a cloud-oriented architecture, the selection of the most relevant supervised learning techniques in the detection of DoS/DDoS attacks, and the implementation of the Intrusion Detection System. The prototype demonstrates that supervised learning techniques can be implemented as cloud services, guaranteeing their performance in detecting these types of attacks in physical networks in real-time.

Keywords: IDS, SOA, Attacks, DoS, DDoS, Machine Learning.

Contenido

Contenido

Lista de figuras

Lista de tablas

Pág.

Lista de abreviaturas

Abreviatura	Término
IDS	Sistema de Detección de Intrusos
SQL	Lenguaje Estructurado de Consulta
SOA	Arquitectura Orientada a Servicio
CNN	Red Neuronal Convolucional
DNN	Red Neuronal Profunda
KNN	K Vecinos Cercanos
RF	Bosque Aleatorio
XGB	Aumento de Gradiente Extremo
DT	Árbol de Decisión
DoS	Ataque de Denegación de Servicios
DDoS	Ataque de Denegación de Servicios Distribuidos
MLP	Perceptrón Multicapa

1 Capítulo 1: Presentación de la Tesis

En el presente capítulo se define el problema y la pregunta de investigación, los objetivos planteados, la metodología establecida y el cronograma. Por último, se establece la estructura del documento y la difusión de los resultados.

1.1 Problema de investigación

La Seguridad Informática es una disciplina que se encarga del bienestar de los activos tecnológicos, en particular haciendo referencia a los datos, información y la infraestructura tecnológica (Figueroa et al., 2018), sin embargo, en (Almanza J., 2019) se mencionan incidentes que afectan la seguridad de los activos, como por ejemplo *phishing*, virus o *malware*, fraude electrónico, inyección SQL y ataques DDoS. Con el fin de dar solución a esta problemática la Seguridad Informática plantea mecanismos de protección, como por ejemplo antivirus, firewalls, VPN, sistemas de detección basados en anomalías, sistemas de prevención y detección de intrusos, como se plantea en (Cano, 2012).

Una de las estrategias planteadas para intentar dar solución a esta problemática, se conoce como Sistema de Detección de Intrusos (IDS); este componente permite detectar el tráfico potencialmente riesgoso en una red de datos (Rojas et al., 2020). El uso de técnicas de Inteligencia Artificial se establece como una propuesta válida para el diseño de Sistemas de Detección de Intrusos, tal como se observa en la tabla 1-1.

Tabla 1-1: IDS Soportados en Técnicas de Inteligencia Artificial

Autores	***Propuesta***	***Conclusión y/o trabajo futuro***
(Pawlicki et al., 2020)	Uso de una red neuronal artificial y el algoritmo Random Forest para la detección de ataques en un dataset previamente establecido.	Implementar mecanismos que permitan reducir la tasa en los falsos positivos.
(Lafram et al., 2019)	Combinación entre algoritmos de Inteligencia Artificial	Se ejecutaron pruebas para definir un análisis comparativo. Como resultado, se obtuvo que los tiempos de ejecución del algoritmo soportado en Redes Neuronales Artificiales fueron los más bajos. Los autores plantean la necesidad de probar otras arquitecturas de red neuronal
(Abughazleh et al., 2019)	Detección de intrusos usando técnicas de interpolación aplicando una red neuronal de función de base radial (RBFNN) como clasificador	Se plantea la necesidad de mejorar los niveles de sensibilidad al momento de detectar ataques de denegación de servicio (DoS) y ataques de respuesta de sonda (probe)

Fuente: Elaboración propia

La implementación de IDS con Redes Neuronales Artificiales (ANN) ha permitido generar un avance en la tasa de detección de intrusos, teniendo en cuenta la capacidad de clasificación de las Redes Neuronales, como se plantea en (Le et al., 2019), (Moukafih et al., 2020) y (Xiao et al., 2019). Las amenazas a las que se enfrentan los sistemas informáticos son diversas tanto por sus características como por el riesgo asociado. Algunas de estas son más relevantes lo que se refleja en las investigaciones y trabajos relacionados. En la tabla 1-2, se recogen trabajos recientes y el tipo de ataque asociado.

Tabla 1-2: Trabajos propuestos sobre detección de ataques de Denegación de Servicios y Denegación de Servicios Distribuidos (DoS/DDoS)

Id	*Año*	*Ataque para detectar*	*Técnica de Inteligencia Artificial usada*	*Autores*
1	2017	DoS	Redes Neuronales Artificiales	Ashfaq et al. (2017)
2	2016	DoS, U2R, R2L, PROBE	Sparse AE	Javaid et al. (2016)
3	2016	DoS, U2R, R2L, PROBE	Redes Neuronales Profundas	Tang et al. (2016)
4	2016	DDoS	Redes Neuronales Artificiales	Saied et al. (2016)
5	2018	DoS, U2R, R2L, PROBE	Redes Bayesianas, Árboles de Decisión	Aljawarneh et al. (2018)
6	2017	DoS, U2R, R2L, PROBE	Redes Neuronales Recurrentes	Yin et al. (2017)
7	2015	DoS, U2R, R2L, PROBE	LSTM + RNN	Kim et al. (2016)
8	2016	DoS DDoS	Redes Neuronales Artificiales	Hodo et al. (2016)
9	2016	DoS, U2R, R2L, PROBE	Redes Neuronales Artificiales y Redes Bayesianas	Pandeeswari y Kumar (2016)
10	2015	DoS	Redes Neuronales Artificiales	Alheeti et al. (2015)
11	2015	DoS, U2R, R2L, PROBE	Red de Crecencia Profunda	Alom et al. (2015)

Fuente: Elaboración propia

En la tabla 1-2 se observa una tendencia asociada a los ataques de Denegación de Servicios (DoS) y Denegación de Servicios Distribuido (DDoS) y tal como se afirma en (Bautista et al., 2018), los ataques DDoS afectan a diferentes infraestructuras tecnológicas. En (Bravo & Mauricio, 2019) se realizó una revisión sistemática de literatura sobre aspectos relevantes en los ataques DDoS, allí se analizaron 96 trabajos sobre métodos científicos para la detección de ataques DDoS, y se concluye que los autores recomiendan la implementación de prototipos de IDS en una red de cómputo, y en particular en equipos servidores, dado el aporte que generar en una red.

El uso de IDS para la mitigación en ataques DDoS se puede observar en (Sallam et al., 2020), (Almseidin & Kovacs, 2019), (Manso et al., 2019) y (Chaudhary & Shrimal, 2019) y en la tabla 1-2 se recogen algunos de estos trabajos. Sin embargo, un espacio abierto está asociado con la construcción de sistemas o herramientas que permitan que los administradores de redes puedan aprovechar los avances en estos trabajos en beneficio concreto de sus instalaciones. Para lograr esto se requiere contar con características que faciliten su uso y garanticen el desempeño del sistema.

Arquitecturas orientadas a servicios que faciliten la integración con otros sistemas, interfaz gráfica y amigable y reportes visualmente claros y dinámicos son algunas de las características deseables. Las arquitecturas orientadas a servicios mejoran la eficiencia en los procesos y reducen la carga en el mantenimiento de sistema y facilitan la adaptación e integración con otros sistemas, según se menciona en (Pantoja et al., 2019).

Las propuestas evaluadas en torno a IDS soportados en técnicas de aprendizaje supervisado, no se orientan a proveer características que faciliten la implementación o replicación por parte de administradores de redes de datos; de tal modo que cuenten con un soporte orientado a detectar y reducir los casos de ataques DDoS.

En vista de la ausencia de propuestas de IDS soportadas en aprendizaje supervisado, orientadas a servicios en la Nube, con fácil acceso al código fuente para reducir los tiempos en la replicación, con características de integración a otros sistemas y que cuenten con interfaz gráfica, generación de reportes y estadísticas, que facilite la administración e implementación por parte de los administradores de redes de datos, se plantean como retos en la presente investigación y que se pueden resumir en:

- Diseño e implementación de la solución IDS basada en aprendizaje supervisado.
- Experimentos con técnicas de aprendizaje supervisado aceptadas por la comunidad científica que resuelvan problemas de intrusiones, para evaluar su desempeño.
- Diseño de la arquitectura orientada a servicios para el IDS.
- Implementación del prototipo computacional de un IDS basado en aprendizaje supervisado en una arquitectura orientada a servicios en la Nube.

De acuerdo con lo anterior y la revisión del estado del arte, se formuló la siguiente pregunta de investigación:

*¿Cómo implementar un Sistema de Detección de Intrusos orientado a servicios en la Nube, usando Aprendizaje Supervisado para la detección de Ataques de Denegación de Servicios Distribuidos**?***

1.2 Objetivos

Objetivo General

Implementar un Sistema de Detección de Intrusos (IDS) soportado en técnicas de aprendizaje supervisado, con una arquitectura orientado a servicios en la Nube para la detección de ataques de Denegación de Servicios Distribuidos (DDoS).

Objetivos Específicos

- Determinar la(s) técnica(s) de aprendizaje supervisado relevantes en detección de ataques DDoS.
- Diseñar la arquitectura del prototipo orientado a servicios.
- Implementar el prototipo computacional de un IDS basado en aprendizaje supervisado en arquitectura orientada a servicios en la Nube.
- Verificación y validación de la implementación orientada a servicios del IDS con un dataset reconocido por la comunidad científica.

1.3 Alcance

- La implementación del sistema de detección de intrusos se enfocará en la detección de ataques de Denegación de Servicios Distribuidos.
- La validación del Sistema de Detección de Intrusos se realizará por medio de *datasets* que hayan sido probados anteriormente.
- El prototipo del sistema de detección de intrusos se realizará bajo una arquitectura orientada a servicios en la Nube y con una interfaz gráfica que genere reportes y estadísticas, buscando que pueda ser aprovechado fácilmente por administradores de red de datos.
- La Tesis no se enfoca a mejorar las métricas de detección de ataques.

1.4 Metodología

Para cumplir los objetivos propuestos se plantean etapas, las cuales están asociadas con los objetivos específicos y se componen de actividades

Tabla 1-3: Metodología

Etapa	Objetivos	Actividades
Etapa 1: Revisión e identificación de las técnicas de aprendizaje supervisado más apropiadas.	**Objetivo 1:** Determinar la(s) técnica(s) de aprendizaje supervisado relevantes en detección de ataques DDoS.	Revisión de técnicas de aprendizaje supervisado usados en la detección de ataques DDoS. Determinar las características de los ataques DDoS. Identificar el funcionamiento de un Sistema de Detección de Intrusos orientado a DDoS.
Etapa 2: Diseño de la arquitectura orientada a servicios.	**Objetivo 2:** Diseñar la arquitectura del prototipo orientado a servicios.	Revisión de literatura sobre marcos de trabajo de software para el desarrollo orientado a servicios. Selección del marco de trabajo orientado a servicios. Definir los componentes y las interacciones en la arquitectura orientada a servicios que soporta la propuesta
Etapa 3: Diseño e implementación del Sistema de Detección de Intrusos propuesto	**Objetivo 3:** Implementar el prototipo computacional de un IDS basado en aprendizaje supervisado en arquitectura orientada a servicios en la Nube**.**	Implementación de la técnica de aprendizaje supervisado seleccionada. Implementación de los componentes en el backend definidos en la arquitectura orientada a servicios en la Nube Implementación de la interfaz gráfica del IDS haciendo uso del marco de trabajo seleccionado.
Etapa 4: Validación del Sistema de Detección de Intrusos	**Objetivo 4:** Verificación y Validación de la implementación orientada a servicios del IDS.	Diseño del ambiente de pruebas para la verificación y validación del Sistema. Aplicación de las pruebas para la verificación y validación.
Etapa 5: Informe final y difusión de los resultados obtenidos en los diferentes medios de divulgación científica.		

Fuente: Elaboración propia

1.5 Alcance

Tabla 1-4: Cronograma de actividades

Actividades	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
Revisión de algoritmos de aprendizaje supervisado usados en la detección de ataques DDoS.	x	x	x									
Determinar las características de los ataques DDoS.	x	x	x	x								
Identificar el funcionamiento de un Sistema de Detección de Intrusos orientado a DDoS.	x	x	x	x								
Revisión de literatura sobre marcos de trabajo de software para el desarrollo orientado a servicios.		x	x	x	x							
Selección del marco de trabajo orientado a servicios.				x	x							
Definir los componentes y las interacciones en la arquitectura orientada a servicios que soporta la propuesta.				x	x	x	x					
Implementación de la técnica de aprendizaje supervisado seleccionada					x	x	x	x				
Implementación de los componentes en el backend definidos en la arquitectura orientada a servicios en la Nube							x	x	x			
Implementación de la interfaz gráfica del IDS haciendo uso del marco de trabajo seleccionado							x	x	x			
Diseño del ambiente de pruebas para la validación del IDS.								x	x			
Aplicación de las pruebas para la verificación y validación.									x	x	x	x

Fuente: Elaboración propia

1.6 Difusión de los Resultados

1.6.1 Participación de Proyectos

"*Prototipo para Detección de Ataques de DDoS (Denegación de Servicios Distribuidos) Basado en Aprendizaje de Máquina en una Arquitectura Orientada a Servicios en la Nube*". **Convocatoria:** "Convocatoria Conjunta de Investigación, Desarrollo Tecnológico e Innovación - 2020".

1.6.2 Eventos

- Primer Simposio Iberoamericano de Maestría y Doctorado en Inteligencia Artificial. Año: 2021.
- Congresso Acadêmico Internacional da Rede Internacional de Autoria Colaborativa (RIAC). Año: 2021
- Sustainable Smart Cities and Territories International Conference - Manizales (Colombia) - 21st-23rd Junio, 2023:
    - Convolutional Neural Network for DDoS Detection.
    - DDoS Attacks Detection with Deep Learning Model using a Cloud architecture.

1.6.3 Publicaciones en Revistas

- Efecto de la Selección de Atributos en el Desempeño de un IDS Basado en Machine Learning para Detección de Intrusos en Ataques DDoS. Revista: South Florida Journal of Development. Estado: Aceptado.
- Feature Selection in the Performance of Machine Learning-based IDS. Estado: En revision.

1.7 Contribuciones

Como contribuciones generadas alrededor de la investigación expuesta en la presente tesis se puede resaltar:

- Acercamiento entra las propuestas de sistemas de detección de intrusos soportadas en aprendizaje supervisado con un enfoque orientado a servicios web.
- Herramienta para administradores de red que no cuentan con recursos para la implementación de sistemas de detección de intrusos soportados en técnicas de aprendizaje supervisado.
- Validación de un prototipo de IDS soportado en aprendizaje supervisado para la detección de ataques DDoS en redes de datos con usuarios y en de tiempo real.
- Análisis comparativo sobre la cantidad de tráfico generada por parte de diferentes tipos de ataques DoS/DDoS.

2 Capítulo 2: Marco teórico

Este capítulo tiene como objetivo presentar los elementos relevantes en el marco de esta Tesis. En la primera sección se presentan aspectos generales relacionados con Seguridad Informática y Seguridad de la Información. En la segunda sección se plantean elementos asociados a los Ataques de Denegación de Servicios (DDoS) y estableciendo su funcionamiento. En la tercera sección se presentan los Sistemas de Detección de Intrusos (IDS), sus características principales, su clasificación y el uso dado por administradores de red. En la siguiente sección se establecen conceptos sobre la Arquitectura Orientada a Servicios (SOA) y sus características. Finalmente se abordan las definiciones asociadas al Aprendizaje Supervisado y sus principales ventajas en términos de clasificación.

2.1 Conceptos Generales sobre Seguridad Informática y Seguridad de la Información

Al momento de hacer referencia a la seguridad en ambientes tecnológicos existe una diferencia entre el alcance y el objetivo de la Seguridad de la Información y Seguridad Informática. La divergencia entre ambos términos se establece en (Figueroa et al., 2018) donde se define el enfoque de la Seguridad Informática y su alcance, el cual está limitado a la protección de los activos en formato digital, mientras que la Seguridad de la Información se encarga de la protección de los activos tecnológicos sin tener en cuenta su estado o forma en que se almacenen.

Según (Sampedro et al., 2019), la Seguridad de la Información puede definirse como un conjunto de acciones que se establecen para asegurar la confidencialidad, integridad y disponibilidad de la información, teniendo en cuenta las diferentes fuentes de almacenamiento, partiendo de la idea de que la información se puede almacenar en diferentes formatos, ya sea en medios digitales o físicos. El conocimiento acerca de la

manera en la que se almacena la información es fundamental para establecer el alcance de políticas de seguridad en ambientes empresariales.

Por otro lado, la Seguridad Informática hace referencia a las medidas que se implementan buscando la protección de los recursos tecnológicos, evitando la divulgación, destrucción o modificación no deseada de la información o componentes técnicos (Humayun et al., 2020). Con relación a esta definición se pueden establecer factores de riesgo que buscan afectar el correcto funcionamiento de un servicio por medio de los recursos informáticos ya sea por hardware o software. En Seguridad Informática se usan de manera frecuente algunos términos relacionados pero que se diferencian entre sí y permiten determinar los niveles de seguridad asociados a un activo (Quiroz & Valencia, 2017), (Humayun et al., 2020):

- **Riesgos:** Es la probabilidad de ocurrencia de un efecto perjudicial e inesperado que afecte un proceso o activo.
- **Amenazas:** Origen de un evento no esperado que causa daños a los recursos informáticos. Estas acciones son realizadas por atacantes que buscan obtener un beneficio sobre las brechas de seguridad de un sistema.
- **Vulnerabilidades:** Es el grado de debilidad que se asocia a un recurso informático y puede ser aprovechado por una amenaza en concreto. Por lo general, una vulnerabilidad se puede asociar a fallos en un sistema, lo que permite que los atacantes puedan ejecutar acciones no permitidas.

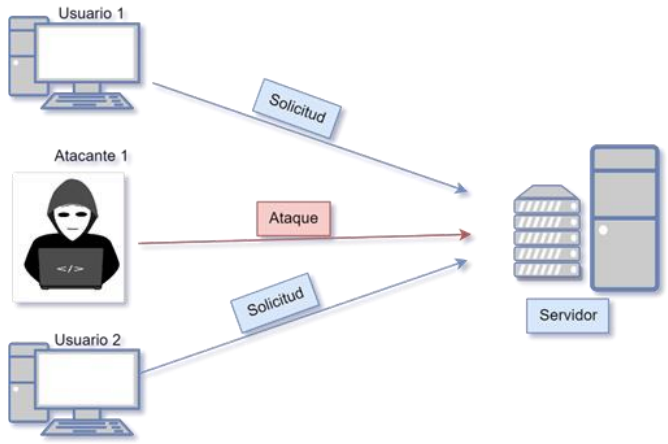
Con relación a las amenazas que se pueden presentar en ambientes informáticos, los ataques de Denegación de Servicios se presentan como un problema que se presenta en la actualidad.

- **Ataques de Tipo DoS/DDoS**

Este tipo de ataques tienen como objetivo principal afectar la disponibilidad de un recurso o servicio en la red, evitando que los usuarios accedan de manera parcial o total. Los ataques de Denegación de Servicios se presentan cuando un atacante envía una gran cantidad de peticiones y el equipo encargado de generar las respuestas no puede procesar de manera exitosa todas las solicitudes recibidas, esto puede provocar colapsos en los

recursos del sistema (Hadeel S. Obaid, 2020). La figura 2-1 representa un ataque de Denegación de Servicios.

Figura 2-1: Ataque de Denegación de Servicios (DoS).



Fuente: Elaboración propia

- **Ataque de Denegación de Servicios Distribuidos (DDoS)**

Se denominan ataques de Denegación de Servicios Distribuidos a los ataques que provienen de muchas fuentes, tal como se observa en la figura 2-2. El origen de los ataques se conoce como equipos zombis, los cuales son simulados por diferentes equipos atacantes. Este tipo de ataques se considera más eficiente con relación a los DoS, teniendo en cuenta la cantidad de equipos que pueden participar en un ataque de manera simultánea. Los ataques DDoS representan una gran amenaza para los Sistemas de Detección de Intrusos, lo que permite que sea un campo de investigación atractivo para la comunidad científica (Hadeel S. Obaid, 2020). Los ataques DDoS pueden ser clasificados principalmente en tres tipos, según lo definido en (Harshita, 2017):

- **Ataques Basados en Volumen:**

En (Harshita, 2017) se establece que en esta categoría se ubican los ataques que buscan saturar el ancho de banda del lado de la víctima. Se caracteriza porque el ancho de banda del equipo atacante supera al ancho de banda del equipo víctima. Se incluyen los ataques ICMP Flood (Inundación del Protocolo de Mensajes de Control de Internet) y UDP (Protocolo de Datagramas de Usuario).

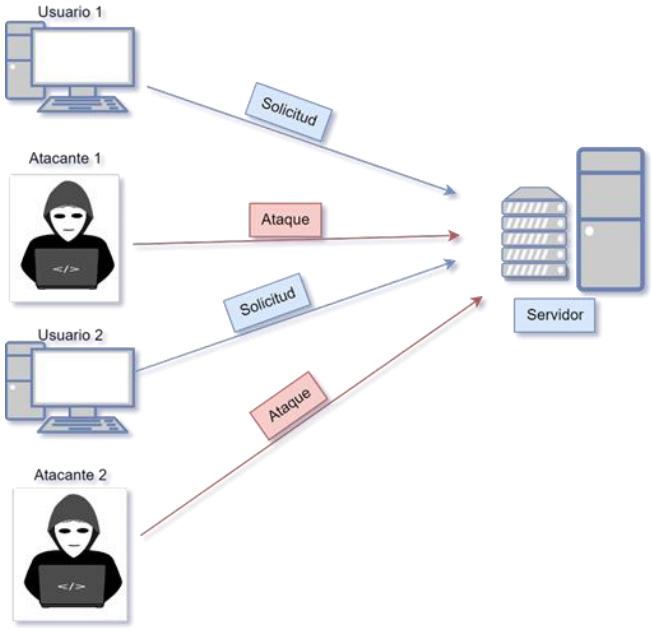
- **Ataques Basados en Protocolo:**

Se incluyen los ataques SYN Flood (Inundación por Sincronización en las Cabeceras del Protocolo de Transporte) y Ping de la Muerte. La principal característica de estos ataques es el consumo excesivo de recursos en la víctima (Harshita, 2017).

- **Ataques a la Capa de Aplicación:**

El objetivo de estos ataques es colapsar los servicios o recursos de la víctima, generando que se afecte la disponibilidad y no estén al alcance de los usuarios legítimos. Estos ataques cuentan con un grado de dificultad alto en términos de detección, y la unidad de medición es el número de peticiones por segundo (Harshita, 2017).

Figura 2-2: Ataque de Denegación de Servicios Distribuidos (DDoS).



Fuente: Elaboración propia

- **Técnicas de Ataques DDoS**

Inundación SYN (SYN Flood): Los ataques por Inundación SYN (Inundación por Sincronización en las Cabeceras del Protocolo de Transporte) consisten en un envío constante de paquetes con peticiones de conexión, las cuales tienen asociadas una dirección IP falsa. El atacante envía la petición de servicio (denominada SYN), el servidor

responde con un SYN/ACK (confirmación de recibido ante la solicitud original), y el cliente debería comunicarse nuevamente con el servidor enviando un ACK (confirmación de recibido). Sin embargo, como la dirección IP es falsa, nunca se genera el ACK de respuesta hacía el servidor, lo que genera un aumento en el número de peticiones recibidas en el servidor y el gasto de recursos computacionales (Solanki & Solanki, 2020).

Los ataques por Inundación SYN se pueden clasificar en 3 tipos (Bogdanoski, 2013):

- **Ataque directo:** Se genera cuando los atacantes envían SYN sin incluir la dirección IP de origen en los archivos de cabecera. Es posible que los atacantes configuren sus equipos de tal modo que no reciban el SYN/ACK por parte del servidor. Esta configuración podría realizarse en el muro cortafuego del sistema operativo del atacante.
- **Ataque envenenado:** En este ataque se realiza una suplantación con direcciones IP válidas, las cuales le permiten al servidor enviar el SYN/ACK a equipos legítimos que no están enviando el SYN, lo cual genera tráfico indeseado.
- **Ataque directo distribuido:** En esta versión del ataque Inundación SYN, el atacante se aprovecha de equipos denominados “zombies” a través de internet. En este caso, cada zombie puede usar múltiples direcciones IP suplantadas, lo que aumenta el rendimiento del ataque.

SMURF: En (Azahari Mohd et al., 2018) se definen los ataques Smurf como el envío de un gran número de paquetes del Protocolo de Mensajes de Control de Internet (ICMP) hacía el equipo de la víctima con mensajes ping falsos.

ICMP Flood: En (Harshita, 2017) los ataques ICMP Flood (Inundación del Protocolo de Mensajes de Control de Internet) se definen como el envío masivo de grandes paquetes de ICMP_ECHO_REQUEST, o comúnmente conocidos como ping. Estos mensajes solicitan una respuesta al equipo receptor, lo que genera una saturación en el ancho de banda que permite la conectividad en el equipo de la víctima.

2.2 Conceptos Generales sobre Sistemas de Detección de Intrusos

Una definición general acerca de un Sistema de Detección de Intrusos es la establecida en (Ahmad et al., 2021a) donde se plantea como un componente dentro del esquema de seguridad informática, cuyo objetivo es la detección de actividades anómalas.

Clasificación de los IDS: En (Ocampo et al., 2017) se realiza una clasificación de los Sistema Detectores de Intrusos así:

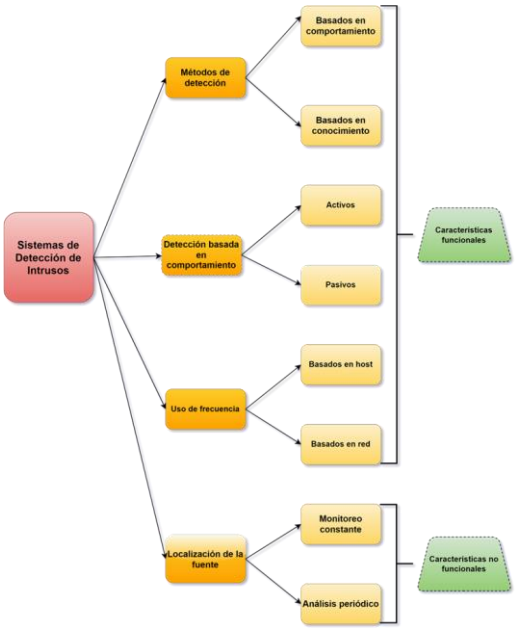
IDS basados en red: Monitorea los paquetes que se detectan en una red determinada para encontrar elementos que indiquen que un intruso ha tenido acceso a la red. Su instalación puede realizarse en un enrutador, para garantizar el análisis de todo el tráfico de la red.

IDS basados en máquinas: Son sistemas encargados de la protección de una máquina en particular, dado que la instalación se realiza en un equipo de estación de trabajo o en un servidor.

IDS basados en periodos o tiempo real: El análisis realizado por este tipo de sistemas varía en función de la frecuencia con la que se ejecute, sin embargo, su finalidad es identificar factores que permitan concluir que existe una intrusión en el sistema o dar respuesta cuando se detecte una anomalía en el sistema en tiempo real.

A continuación, en la figura 2-3 se puede observar una clasificación para los Sistemas de Detección de Intrusos.

Figura 2-3: Resumen- sobre los Sistemas de Detección de Intrusos



Fuente: Elaboración propia

2.3 Arquitectura de Software Orientada a Servicios (SOA)

La Arquitectura Orientada a Servicios es un concepto que se asocia en ambientes empresariales y en la relación que existe con el área de Tecnologías de Información y Comunicaciones, sin embargo, una arquitectura orientada a servicios (SOA) no se enmarca totalmente dentro de una tecnología o servicio, tal como se menciona en (Niknejad et al., 2020). Allí los autores resaltan que una arquitectura SOA puede aplicarse como un patrón arquitectónico que se alinea con la idea de negocio a partir de soluciones empresariales (Lublinsky, 2007).

La implementación de SOA en ambientes empresariales permite aumentar las interacciones dinámicas entre sistemas no integrados directamente, lo que reduce el nivel de acoplamiento y facilita la escalabilidad entre los sistemas (Alonso et al., 2018).

Algunas de las ventajas de la implementación de arquitecturas orientadas a servicios se pueden observar en la tabla 2-1.

Tabla 2-1: Aplicaciones de SOA

Autor	Aplicación	Conclusión
(Balaji et al., 2018)	Adaptabilidad de SOA en Servicios de Internet de las Cosas	Análisis sobre la arquitectura SOA para tecnologías de Internet de las Cosas y su integración con servicios web.
(Arango Serna et al., 2010)	Arquitectura empresarial	La lógica empresarial se divide en unidades más pequeñas y fáciles de programar.
(Chang et al., 2022)	Sistemas de detección de intrusos para computación en la nube	En este artículo presentan un resumen sobre las diferentes propuestas de sistemas de detección de intrusos que están soportados en la nube.

Fuente: Elaboración propia

2.4 Conceptos Generales sobre Aprendizaje Supervisado

El Aprendizaje Supervisado es una agrupación del Aprendizaje de Máquina (*Machine Learning*) el cual permite detectar un comportamiento normal o atípico en un conjunto de datos (Maldonado, 2018). Una de las características más importantes del Aprendizaje Supervisado es la posibilidad de generar predicciones a partir de cálculos matemáticos, donde se conoce cuál es la característica de un objeto y se conoce su clase (Correa Wachter & Henao Villas, 2021).

Métricas para Evaluación de Rendimiento

Para evaluar el rendimiento de los diferentes clasificadores existen métricas fundamentales que aportan información relevante sobre la predicción de los modelos. El análisis de las métricas puede constituir una evaluación sencilla ante las predicciones de las técnicas. A continuación, se presentan las fórmulas para evaluar el rendimiento de los modelos.

I. **Exactitud (Accuracy):** Se define como la cantidad de predicciones correctas sobre el total de predicciones. Se recomienda usar como un parámetro confiable para la evaluación de un modelo cuando los datos se encuentran equilibrados, dado que una clase mayoritaria podría afectar el rendimiento del modelo (Mishra & Pandya, 2021).

Fórmula 1: Exactitud

$$\boldsymbol{Exactitud} = \frac{(\boldsymbol{TP} + \boldsymbol{TN})}{(\boldsymbol{TP} + \boldsymbol{TN} + \boldsymbol{FP} + \boldsymbol{FN})}$$

Fuente: (Disha & Waheed, 2022)

Donde:

- ***TP:*** Verdadero Positivo (*True Positive*):
- ***TN:*** Verdadero Negativo (*True Negative*).
- ***FP:*** Falso Positivo (*False Positive*).
- ***FN:*** Falso Negativo (*False Negative*).

II. **Sensibilidad / Exhaustividad (*Recall*):** También se denomina tasa de detección, e indica la relación entre todos los registros clasificados como ataques y que son realmente ataques (Ahmad et al., 2021b).

Fórmula 2: Sensibilidad

$$\boldsymbol{Sensibilidad} = \frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FN})}$$

Fuente: (Disha & Waheed, 2022)

III. **Precisión (*Precision*):** Indica el número de total de predicciones positivas las que eran en realidad positivas con el número total de positivos reales (Gangula et al., 2022).

Fórmula 3: Precisión

$$\boldsymbol{Precisión} = \frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FP})}$$

***Fuente*:** (Disha & Waheed, 2022)

IV. **Puntaje F1 (*F1 score*):** Métrica de evaluación usada en conjuntos de datos desequilibrados, la cual se calcula a partir de la media armónica de la precisión (Mishra & Pandya, 2021).

Fórmula 4: Puntaje F1 (F1 score)

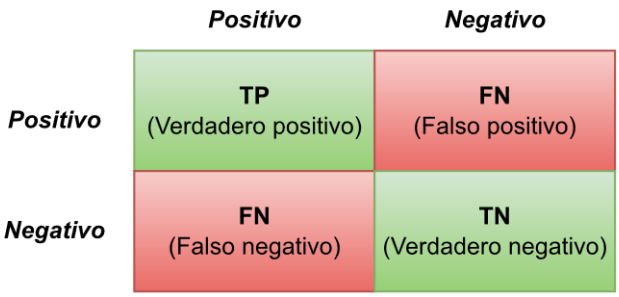
$$\boldsymbol{Puntaje\ F1} = \frac{(\mathbf{2} * \boldsymbol{Precisión} * \mathbf{Sensiblidad})}{(\boldsymbol{Precisión} + \boldsymbol{Sensibilidad})}$$

***Fuente*:** (Disha & Waheed, 2022)

V. Matriz de Confusión

Una matriz de confusión es una herramienta usada con frecuencia en el aprendizaje automático supervisado para medir el comportamiento de los modelos de clasificación (Hasnain et al., 2020). En la figura 2-4 se observa la distribución de los falsos negativos, falsos positivos, verdaderos positivos y verdaderos negativos en su forma matricial.

Figura 2-4: Matriz de Confusión



***Fuente*:** (Mishra & Pandya, 2021)

Tal como se observa en la figura 2-4, lo ideal para el rendimiento de un modelo de clasificación desde un análisis en una matriz de confusión es que todos los elementos estén ubicados en la diagonal principal de la matriz.

Técnicas de Aprendizaje Supervisado

K Vecinos Cercanos (K *Nearest Neighbor* - KNN):

Es un algoritmo de clasificación ampliamente usado en tareas de clasificación, tal como se menciona en (Wu et al., 2008). Su principal característica es que no requiere grandes volúmenes de datos para su entrenamiento, sin embargo, su costo está en la etapa de validación. Esta técnica clasifica de acuerdo con los siguientes pasos (Beckmann et al., 2015):

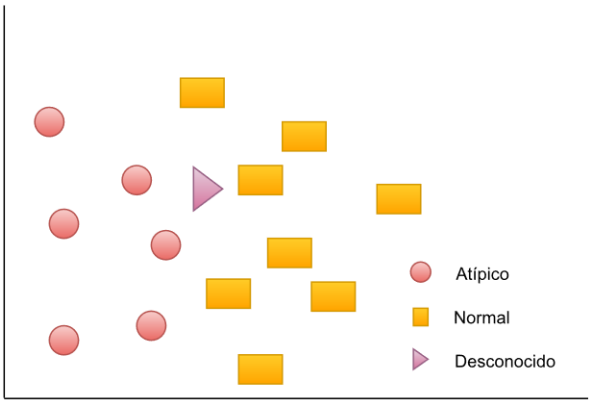
- Cálculo de la distancia euclidiana desde un registro hasta el resto de los valores del conjunto de entrenamiento.
- Selección de k elementos vecinos más cercanos.
- Al registro se le asigna una etiqueta con la misma clase de los k vecinos más próximos.

En (Susa Velandia et al., 2022) se define como una de las ventajas del clasificador KNN es su facilidad de implementación, no obstante, su principal debilidad es el tiempo de

predicción, dado que debe realizarse el cálculo de la distancia entre todos los puntos de datos.

Tal como se observa en la figura 2-5, los círculos rojos representan los valores atípicos en un conjunto de datos, mientras que los rectángulos de color amarillo se clasifican como valores normales

Figura 2-5: Principio de clasificación de KNN



***Fuente*:** (Asharf et al., 2020).

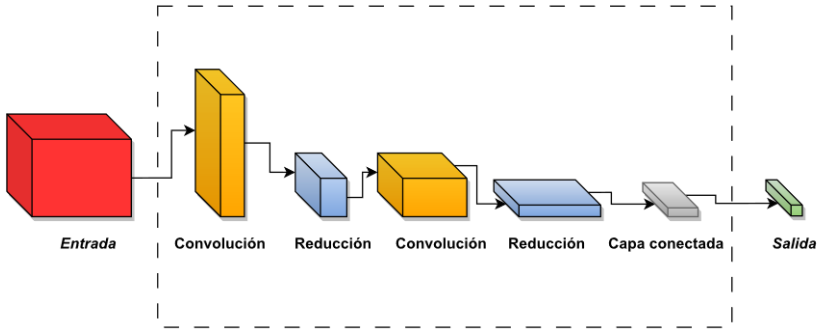
Para clasificar los valores no conocidos, en este caso, el triángulo de color morado, se calcula la distancia entre dicha figura y sus k vecinos cercanos. Para K=1, el triángulo será clasificado como normal, sin embargo, si K=2 o K=3, el triángulo sería clasificado como atípico (Asharf et al., 2020).

Red Neuronal Convolucional (Convolutional Neural Network – CNN):

En (Snigdho et al., 2022) se define a una Red Neuronal Convolucional como un tipo de algoritmo de aprendizaje profundo (*deep learning*), el cual puede obtener predicciones a partir de datos de entrada por medio de capas ocultas, entre las cuales se encuentran las capas de convolución, reducción y capa conectada hasta obtener la salida, tal como se observa en la figura 2-6.

Este tipo de redes son ampliamente usadas dado su alto rendimiento y son consideradas las más precisas, según lo expuesto en (Issa et al., 2023).

Figura 2-6: Arquitectura Red Neuronal Convolucional



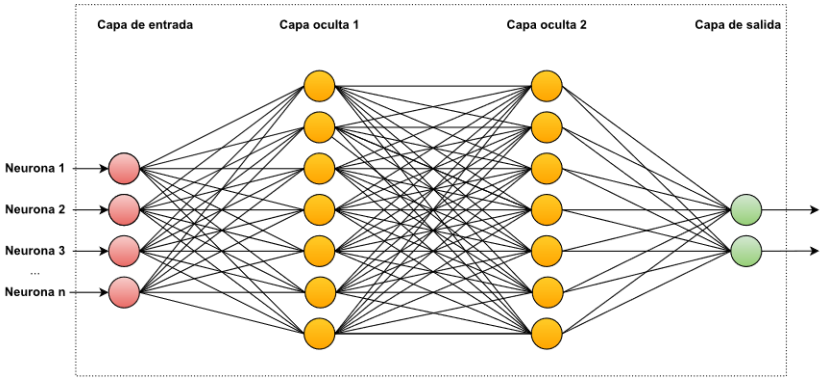
***Fuente*:** (Andrade Carrera et al., 2021)

Tal como se observa en la figura 2-6, las capas de convolución aplican filtros a los datos de entrada, mientras que las capas de reducción se encarga principalmente aumentar la velocidad en el proceso de entrenamiento y minimizar los parámetros para evitar sobreentrenamiento (Mohammadpour et al., 2022).

Red Neuronal Profunda (*Deep Neural Network – DNN*):

Una Red Neuronal Profunda es un tipo de red neuronal ampliamente usada en diferentes tareas de clasificación, como el reconocimiento de imágenes y robótica. Este tipo de redes pueden tener varias capas ocultas y en cada capa un número determinado de neuronas, las cuales realizan cálculos a partir de los valores de entrada (capa de entrada) (Sze et al., 2017). A continuación, se define en la figura 2-7, una arquitectura simple para una Red Neuronal Profunda.

Figura 2-7: Arquitectura Red Neuronal Profunda



***Fuente*:** (Merchán & César, 2022)

Tal como se observa en la figura 2-7, esta Red Neuronal Profunda cuenta con una arquitectura genérica definida así (***n,m,m,2***). Siendo n el número de neuronas de la capa de entrada, m el número de neuronas de las capas ocultas y 2 para definir las salidas de la red.

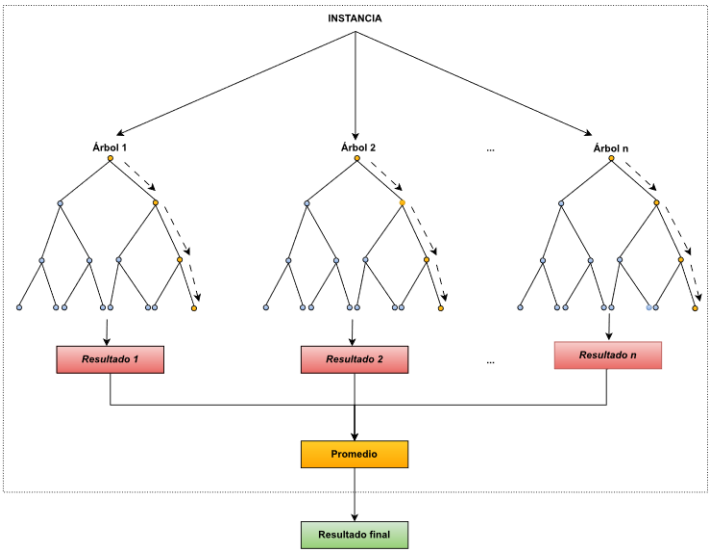
En (K. J. Lee, 2021) se resalta que en ocasiones es necesario buscar estrategias para optimizar el comportamiento de la red neuronal sin comprometer las métricas, dado que este tipo de redes no son tan sencillas como lo son las MLP.

Bosque Aleatorio (Random *Forest):*

El clasificador Bosque Aleatorio es una técnica de aprendizaje supervisado que genera árboles de decisión para la fase de entrenamiento. La división de cada subnodo se realiza aleatoriamente para minimizar el sobreentrenamiento (W. Wang et al., 2021).

Tal como se observa en la figura 2-8, el resultado final se define por medio del cálculo del promedio de todos los subconjuntos de árboles.

Figura 2-8: Arquitectura Bosque Aleatorio



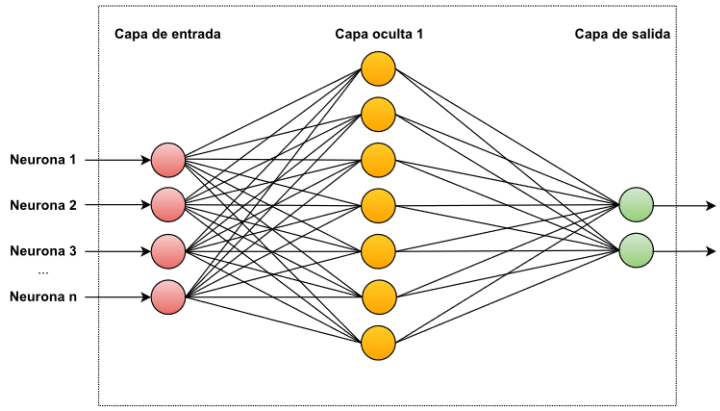
***Fuente*:** (W. Wang et al., 2021)

Red Neuronal Perceptrón Multicapa (Multilayer *Perceptron – MLP):*

Una Red Neuronal Perceptrón Multicapa se define en (Quirumbay Yagual et al., 2022) como un algoritmo de clasificación ubicado entre las técnicas de aprendizaje supervisado la cual se compone de una capa de neuronas de entradas, una capa de neuronas de

procesamiento y la capa de neuronas de salida, tal como se observa en la figura 2-9. En una red de tipo MLP cada nodo (neurona) se conecta con por medio de un peso calculado a través de una función de activación a otro nodo en la capa siguiente.

Figura 2-9: Arquitectura Red Neuronal Perceptrón Multicapa



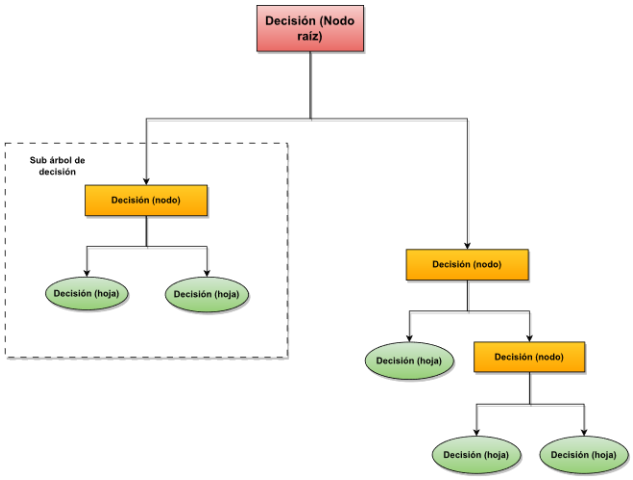
***Fuente*:** (Quirumbay Yagual et al., 2022).

Las redes neuronales MLP son consideradas sensibles en el escalado de atributos, por lo cual es fundamental realizar diferentes cálculos utilizando diferentes hiperparámetros, no obstante, esto puede elevar la complejidad computacional de los modelos (Quirumbay Yagual et al., 2022).

Árboles de Decisión (*Decision Tree - DT*):

El árbol de decisión es una técnica de clasificación no paramétrica que genera predicciones a partir del uso de reglas basadas en la agrupación de criterios (Layme Fernández et al., 2022). En (Caballero et al., 2019) se define el inicio de un árbol de decisión bajo un único nodo, lo que permite una ramificación a partir de los resultados. Otra característica importante es que a medida que aumenta el número de nodos, también se incrementa el número de posibles resultados finales para el modelo predictivo. A continuación, se expone en la figura 2-10, la estructura genérica de un Árbol de Decisión.

Figura 2-10: Estructura genérica de un Árbol de Decisión



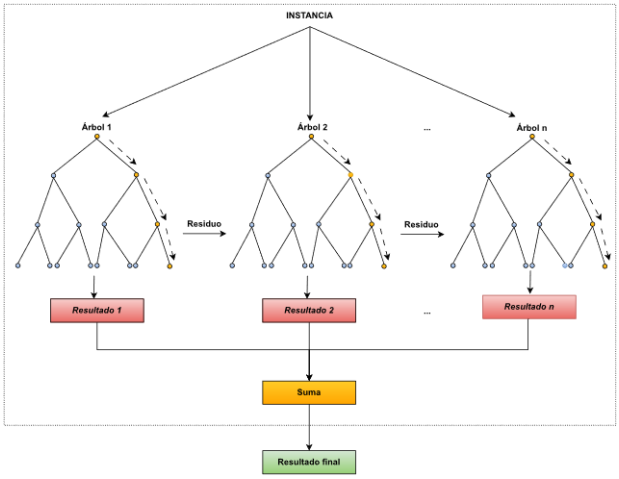
***Fuente*:** Elaboración propia

Aumento del Gradiente Extremo (*XGB):*

Es una librería implementada a partir de las técnicas de clasificación Gradient boosting o Potenciación del gradiente la cual tiene un comportamiento similar al de Random Forest desde (W. Wang et al., 2021).

Tal como se observa en la figura 2-11, el resultado generado en el subconjunto del árbol 1, es la entrada para el siguiente subconjunto. Para determinar una predicción, el algoritmo suma cada uno de los resultados obtenidos por los subconjuntos definidos en el modelo.

Figura 2-11: Arquitectura Aumento del Gradiente Extremo

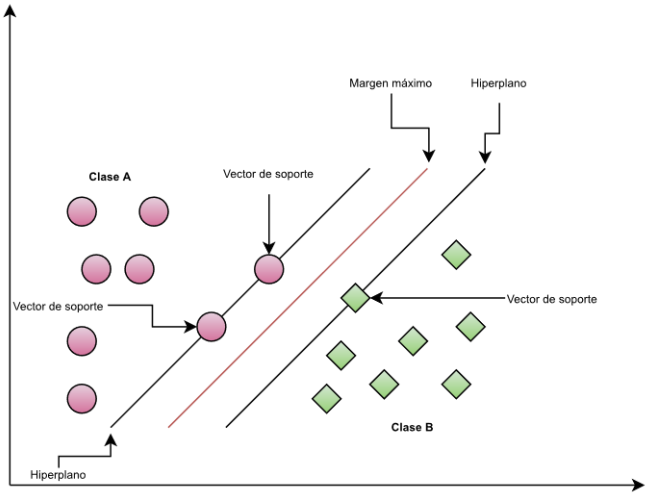


***Fuente*:** (W. Wang et al., 2021).

Máquina de Soporte Vectorial (*Support Vector Machine - SVM*):

Una Máquina de Soporte Vectorial es una técnica de clasificación que pertenece al aprendizaje supervisado que analiza datos para determinar un patrón y obtener una clasificación. En un principio se consideraba sólo para problemas binarios, sin embargo, su uso se ha extendido en tareas de clasificación multiclase. SVM es un clasificador lineal que no necesita grandes cantidades de datos para aprender patrones y generar predicciones (Rt' et al., 2014). En la figura 2-12 se observan algunos elementos fundamentales para comprender un SVM, los vectores de soporte e hiperplanos.

Figura 2-12: Arquitectura Máquina de Soporte Vectorial



***Fuente*:** Elaboración propia

Se denominan Vectores de Soporte a los puntos más cercanos a la línea que genera el margen de separación, a esta línea se le conoce como hiperplano.

2.5 Conceptos Adicionales

2.5.1 Conjunto de datos CICIDS2017 (*dataset* CICIDS2017):

El conjunto de datos CICIDS2017 es un dataset elaborado por investigadores del Instituto Canadiense para la Ciberseguridad, el cual fue diseñado para ayudar a la creación de modelos para la detección de ataques informáticos (Panigrahi & Borah, 2018). Según lo

expuesto en (Sharafaldin et al., 2018) el dataset tiene las características expuestas en la tabla 2-2.

Tabla 2-2: Dataset CICIDS2017

Día de Actividad	Clasificación
Lunes	Tráfico benigno
Martes	Tráfico benigno, Ataque FTP - Patator, Ataque SSH-Patator
Miércoles	Tráfico benigno, Ataque DoS GoldenEye, Ataque DoS Hulk, Ataque DoS Slowhttptest, Ataque DoS slowloris, Heartbleed
Jueves	Tráfico benigno, Ataque de Fuerza Bruta – Inyección SQL, Ataque XSS
Jueves	Tráfico benigno, Infiltración
Viernes	Tráfico benigno, Bot
Viernes	Tráfico benigno, Escaneo de puertos
Viernes	Tráfico benigno, Ataque DDoS

***Fuente*:** Elaboración propia

En la tabla 2-3, se observa la distribución de las clases y el número total de registros por cada categoría.

Tabla 2-3: Clases y total de instancias en dataset CICIDS2017

Etiquetas de las clases	Número de instancias
Beningo	2359087
DoS Hulk	231072
PortScan	158930
DDoS	41835
DoS GoldenEye	10293
FTP-Patator	7938
SSH-Patator	5897

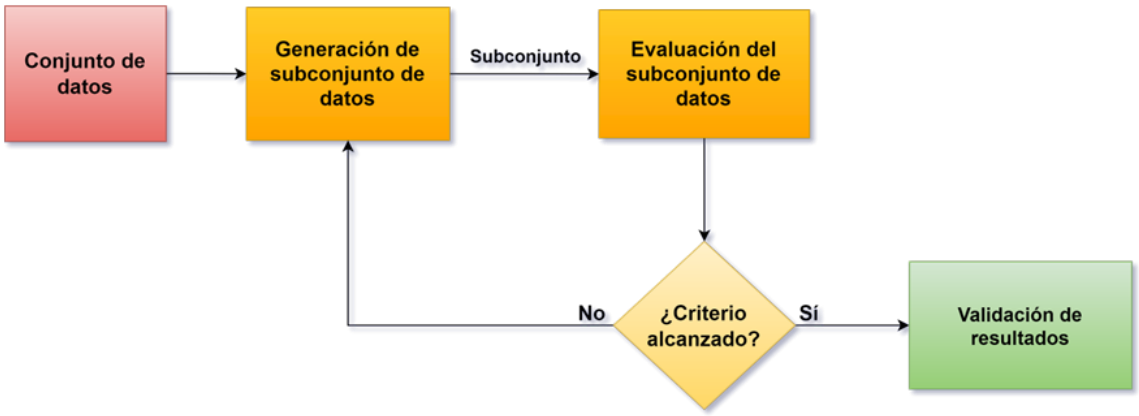
***Fuente*:** (Sharafaldin et al., 2018)

Tal como se observa en la tabla 2-3, el dataset CICIDS2017 registra diferentes tipos de ataques, no obstante, esta tesis se centra sólo en los ataques DoS/DDoS. Este dataset contiene un total de 79 columnas, de las cuales se debe resaltar que una de ellas se repite.

2.5.2 Selección de Características

La selección de características es una estrategia implementada en conjuntos de datos grandes que busca generar una reducción en la dimensionalidad de un problema, de tal forma que se pueden obtener mejores resultados en términos de costo computacional (Iram et al., 2020).

Figura 2-13: Procedimiento general para aplicar selección de características



Fuente: (Kumar, 2014)

El procedimiento observado en la figura 2-13 se define como una estrategia genérica que puede ser aplicado a diferentes técnicas de selección de características, incluyendo Boruta, importancia de características, análisis de componentes principales (PCA), Chi-Cuadrado, entre otros, tal como se menciona en (Thakkar & Lohiya, 2021), (Krishnaveni et al., 2021) y (Farhana et al., 2022).

2.6 Conclusiones del capítulo

En este capítulo se definieron los fundamentos de los principales elementos a tener en cuenta en el marco de esta tesis. En primera instancia, se resaltaron conceptos sobre seguridad informática, y seguridad de la información, resaltando los ataques de tipo DoS / DDoS. En segunda instancia, se definieron los conceptos acerca de los Sistemas de Detección de Intrusos, Arquitectura Orientada a Servicios y Aprendizaje Supervisado. En

última instancia, se añadieron conceptos adicionales sobre el conjunto de datos CICIDS2017 y selección de características.

3 Capítulo 3: Estado del Arte

En este capítulo se presenta una revisión sistemática de literatura sobre los principales trabajos publicados en torno a los Sistemas de Detección de Intrusos basados en Técnicas de Aprendizaje Supervisado para la detección de Ataques de Denegación de Servicios Distribuidos, describiendo los enfoques de solución y exponiendo las fortalezas y debilidades. En primer lugar, se describe la metodología empleada para la búsqueda sistemática, en segundo lugar, se analiza cada uno de los trabajos relacionados y finalmente se exponen las conclusiones asociadas a la problemática

3.1 Revisión Sistemática de Literatura

El proceso de revisión sistemática de literatura se realizó por medio de una cadena de búsqueda con palabras claves en inglés. En la tabla 3-1 se observa la ecuación construida.

Tabla 3-1: Ecuación de búsqueda

Ecuación de búsqueda	(Security OR Attacks OR Cybersecurity)
	AND
	(Supervised Learning OR "SL")
	AND
	(Intrusion Detection System OR "IDS")
	AND
	(DDoS Attacks)

***Fuente*:** Elaboración propia

La ecuación de búsqueda fue ejecutada el día 12 de diciembre de 2022 en Scopus, Science Direct y en IEEE Xplore. Estas bases de datos se seleccionaron teniendo en cuenta su alto impacto académico.

Como criterios de inclusión se tuvieron en cuenta:

- Año de publicación (trabajos publicados desde el año 2020).
- Idioma (inglés).
- Tipo de publicación (artículos de investigación).
- Estado (disponibilidad del documento).
- Área temática (ingeniería).

Por otro lado, se consideró una revisión narrativa que arrojó un total de 12 trabajos. En la Tabla 3-2 se observa la cantidad total.

Tabla 3-2: Trabajos relacionados

Base de datos	Cantidad
Scopus	2
Science Direct	42
IEEE Xplore	18
Total de trabajos desde la revisión sistemática	62
Total de trabajos desde la revisión narrativa	12
TOTAL	74

***Fuente*:** Elaboración propia

Algunos de los trabajos encontrados no fueron tenidos en cuenta para la construcción del presente capítulo, dado que el enfoque que allí se presenta difiere con la orientación de la investigación definida en esta Tesis.

3.2 Discusión de Trabajos Relacionados

En (Doriguzzi-Corin et al., 2020) se presenta un sistema de detección DDoS soportado en Redes Neuronales Convolucionales (CNN) denominado LUCID. Los autores plantean como contribución a la comunidad científica el enfoque innovador de las CNN para la detección de intrusiones y una validación de su propuesta en plataformas computaciones de bajos recursos. Según lo reportado en los resultados, se evidencia una mejora de hasta 40 veces en el tiempo de procesamiento con respecto a las demás propuestas encontradas en el estado del arte, sin embargo, se plantea como trabajo futuro la revisión de los falsos positivos obtenidos en los resultados.

Fortalezas: Evaluación de la propuesta con diferentes conjuntos de datos (3 en total).

Debilidades: No hay validación con tráfico en tiempo real ni con usuarios.

En (Bhardwaj et al., 2020) se propone una arquitectura de Red Neuronal Profunda usando AutoEncoder para clasificar el tráfico benigno y el tráfico asociado a los ataques DDoS. En este trabajo los autores realizaron una comparación entre 10 enfoques soportados en características de *Deep Learning* encontrados en la literatura científica, y allí se establecieron parámetros para analizar el rendimiento, tales como la precisión en la detección, exhaustividad y exactitud. El trabajo superó los enfoques encontrados usando el dataset NSL-KDD y resultados competitivos con respecto al uso del dataset CICIDS2017. Como trabajo futuro se plantea mejorar el rendimiento en detección de intrusos sobre el tráfico en tiempo real y fortalecer la validación de la propuesta usando más conjuntos de datos.

Fortalezas: Validación con el dataset CICIDS2017 y NSL-KDD.

Debilidades: No se generó validación con tráfico en tiempo real.

En (Phan et al., 2020) se realizó una propuesta para la detección de ataques DDoS sobre redes definidas por software (SDN) denominado DEEPGUARD. Con respecto a los resultados obtenidos, se evidencia una mejora en cuanto al rendimiento en comparación con otros métodos de detección de ataques de denegación de servicios distribuidos en SDN. Los autores plantean la necesidad de probar este sistema para otros tipos de ataques informáticos.

Fortalezas: Evaluación de la propuesta usando SDN y análisis detallado a diferentes tipos de ataque DDoS (*Slowloris y SYN Flood*).

Debilidades: Enfoque exclusivo a los ataques de tipo DDoS.

En (Haider et al., 2020) los autores proponen una Red Neuronal Convolucional Profunda (CNN) para la detección de ataques DDoS en SDN. Con respecto a los resultados, los autores plantean una mejora en la exactitud, exhaustividad, precisión y tiempos de entrenamiento en comparación con otros trabajos relacionados. Se resalta la importancia de los diversos mecanismos de detección y prevención de intrusos que se soportan en *Deep Learning*, teniendo en cuenta los avances que se vienen presentando en el campo de las telecomunicaciones.

Fortalezas: Evaluación de la propuesta usando Redes Definidas por Software.

Debilidades: No se realizó una validación de la propuesta en una subred en tiempo real con usuarios.

En (Čelesová et al., 2019) la propuesta realizada por los autores se establece como un sistema de detección de intrusos basado en una red neuronal profunda (NIDS-DNN). Los parámetros de evaluación del rendimiento del NIDS fueron la precisión, sensibilidad, especificidad y exactitud. El enfoque definido en este trabajo fue para redes definidas por software, y se tomó como medida de evaluación un total de 100 usuarios, de los cuales 80 eran benignos y 20 eran maliciosos. Como trabajo futuro se plantea un análisis con el dataset UNSW-NB15 para comparar los resultados obtenidos y mejorar el rendimiento.

Fortalezas: Validación con varios conjuntos de datos (KDD CUP 99, UNSW-NB15).

Debilidades: No se realizó una comparación de métricas de rendimiento desde otras técnicas.

En (Virupakshar et al., 2020) se propone analizar el tráfico para la detección de DDoS en ambientes virtualizados. Como apoyo para la detección de los ataques, los autores usaron algoritmos de clasificación, entre los cuales se encuentran Árboles de Decisión, K vecinos más próximos (KNN), Clasificadores Bayesianos y DNN. Los autores hacen énfasis en la necesidad de diseñar estrategias asociadas a los algoritmos de clasificación para obtener mejores resultados en los parámetros típicos de rendimiento y aumentar los niveles de detección en los ataques.

Fortalezas: Notificación en caso de la detección de un tráfico anómalo y análisis con diferentes técnicas de clasificación. Adicionalmente, se realizó un análisis con tráfico real.

Debilidades: Si bien los autores realizaron validación con subred en tiempo real, no se ejecutaron pruebas para determinar la viabilidad de la propuesta con usuarios en la red.

En (T. H. Lee et al., 2020) los autores presentan un Sistema de Detección y Prevención de Intrusos (IDPS) para la prevención de ataques de fuerza bruta y denegación de servicios distribuidos para redes definidas por software. Allí se comparan Redes Neuronales Convolucionales, Perceptrones Multicapas (MLP), Memoria a Largo Plazo (LSTM) y el

Codificador Automático Apilado (SAE). Los resultados muestran que el IDPS soportado en MLP alcanza una precisión del 99% en la prevención de ataques de fuerza bruta y casi el 100% en prevención de ataques DDoS. Como trabajos futuros, los autores plantean la necesidad de realizar pruebas en tiempo real, dado que los resultados obtenidos por el IDPS pueden cambiar en función del flujo de los datos.

Fortalezas: Comparación del rendimiento de la CPU con propuesta de IDS y Firewall para la detección de ataques DDoS.

Debilidades: Los autores no realizaron una validación del funcionamiento de su propuesta con tráfico en tiempo real.

En (Roopak et al., 2020) se plantea un marco de trabajo basado en redes neuronales convolucionales profundas para la detección de ataques DDoS para redes definidas por software. Se realiza una comparación entre parámetros de rendimiento como exhaustividad, precisión y tiempo de procesamiento para los trabajos encontrados en el estado del arte. Los autores apoyan la creación de nuevas iniciativas soportadas en redes neuronales profundas para detección de ataques DDoS, teniendo en cuenta el aumento de las redes a gran escala y el incremento en los ataques de este tipo.

Fortalezas: Aplicación de selección de características y validación con varios conjuntos de datos (CICIDS2017, UNBISCX, SDN Collected, KDDCup99).

Debilidades: Ausencia de validación con todo el conjunto de atributos completo, dado que se consideró reducción de atributos.

En (Moukhaf et al., 2019) se implementa una red neuronal artificial optimizada por medio de la técnica de algoritmos genéticos buscando establecer un modelo para la detección de intrusos en un sistema. Se usó el dataset KDD99; cuyos datos son procesados frecuentemente en estudios relacionados con temáticas de IDS. En el apartado de trabajos futuros, los autores plantean la necesidad de implementar estrategias de conglomerado para garantizar la calidad de los datos y obtener mejores resultados con respecto a la predicción del sistema.

Fortalezas: Validación con varios conjuntos de datos (UNSW-nb15, KDD99) y clasificación de diferentes tipos de ataques.

Debilidades: No se detallan las métricas en fase de prueba y de entrenamiento de su propuesta.

En (Taheri et al., 2018) se propone un motor basado en *Deep Learning* para la detección de robots que propagan los ataques DDoS por medio de las imágenes generadas por el tráfico en una red de datos. El sistema implementado en este trabajo transforma los datos del tráfico de la red en imágenes que posteriormente son analizadas por la red convolucional profunda, usando el dataset CTU-13. Como trabajo futuro, se plantea la detección de otros tipos de ataques en seguridad informática además del uso de otros tipos de redes neuronales para establecer parámetros de comparación.

Fortalezas: Análisis del tráfico en tiempo real.

Debilidades: El estudio no presenta una reducción de características al conjunto de datos para evaluar la viabilidad de la detección de ataques con un número de características reducida.

En (Shi et al., 2019) se propuso un método denominado *DeepDDoS*, soportado en *Deep Learning* para la detección de ataques de denegación de servicios distribuidos usando el dataset CICIDS2017. La detección se realiza analizando periodos de tiempo y paquetes del flujo de datos. Los autores plantean que la detección de ataques por medio agrupaciones de máximo 5 paquetes puede generar un mejor rendimiento en cuanto a los niveles de latencia. Como trabajo futuro, los autores plantean la necesidad de generar pruebas con otros conjuntos de datos que contengan firmas de ataques DDoS, para analizar el rendimiento del sistema en relación con las tasas de detección.

Fortalezas: Validación de la propuesta con un dataset reconocido por la comunidad científica (CICIDS2017). Los autores realizaron una evaluación del rendimiento con diferentes técnicas de clasificación usando tráfico en tiempo real y generando una alerta en caso de que fuera detectado un ataque.

Debilidades: No se proporcionan mecanismos para generar reportes de acuerdo con la detección de tráfico DDoS.

(Spiekermann & Keller, 2021) proponen una investigación orientada a la detección de anomalías basada técnicas de aprendizaje no supervisado, IsolationForest y

LocalOutlierFactor. El análisis realizado por los autores se estableció bajo el uso de Redes Definidas por Software (SDN), en concreto, por medio de una Virtual LAN (VLAN). Los autores definieron un entorno virtual basado en OpenStack, una suite de servicios de computación en la nube. Los autores resaltan la flexibilidad que pueden tener las redes virtuales en términos de administración, sin embargo, al utilizar técnicas de Aprendizaje de Máquina, cualquier modificación en la infraestructura de red puede alterar el rendimiento de los algoritmos. Algunas alteraciones como la modificación de usuarios o creación de nuevas máquinas virtuales pueden generar cambios en detección de anomalías. Por último, se hace énfasis en la importancia de entrenar los modelos de clasificación usando conjuntos de datos válidos y actualizados, para obtener las características recientes de las redes modernas. Como trabajo futuro se define la validación de la propuesta con grandes capturas de tráfico bajo el formato del *dataset* UNSW-NB15.

Fortalezas: Validación de la propuesta con algoritmos diferentes a los comúnmente definidos en la literatura (IsolationForest y LocalOutlierFactor.

Debilidades: Ausencia de un análisis con métricas como precisión, puntaje f1 y exhaustividad. Los autores no validaron su propuesta con el dataset CICIDS2017.

En (Dong et al., 2022) los autores proponen la detección de tráfico anómalo en una red por medio de un autocodificador variacional, usando el dataset NSL-KDD. El modelo que allí se propone está compuesto de 2 etapas. La primera consistió en el preprocesamiento de los datos, y la segunda en la predicción por medio del modelo. Los autores realizaron comparaciones en el rendimiento de la precisión, exhaustividad, puntaje F1 y exactitud entre los clasificadores Bosque Aleatorio (RF), Máquina de Soporte Vectorial (SVM), Regresión Logística (LR), Perceptrón Multicapa (MP), Autocodificador Variacional (VAE) y Autocodificador Variacional de Condicionales Avanzadas (A-CVAE). Se concluye que el método A-CVAE arrojó los mejores resultados durante las pruebas realizadas. Adicionalmente, se recomienda aumentar el número de instancias para el conjunto de entrenamiento, si el objetivo es aumentar la precisión del modelo (87.27%).

Fortalezas: Proceso de entrenamiento con un conjunto de datos reconocido por la comunidad (NSL-KDD).

Debilidades: No validación con tráfico en tiempo real.

En (Gangula et al., 2022) se define un sistema que identifica y clasifica ataques de tipo TCP-SYN, UDP Flood, ICMP-echo, HTTP Flood, Slow Loris, Slow Post y *Brute Force,* definidas en el dataset SNMP-MIB. En primera instancia se realiza el preprocesamiento de los datos, en donde fueron eliminados datos atípicos. En segunda instancia, los registros son codificados en valores numéricos. En tercera instancia, se realiza una selección de características para reducir la dimensionalidad del dataset y por último se define la clasificación usando el algoritmo Red Neuronal de Unidades Recurrentes con Compuerta Basada en el Promediado Bidireccional Ponderado de Características (GRU-BWFA). Los autores resaltan la dificultad que representa la categorización de ataques DDoS, lo que puede generar clasificaciones incorrectas en diferentes modelos.

Fortalezas: Clasificación de diferentes tipos de ataques DDoS.

Debilidades: Evaluación con otras técnicas de clasificación ampliamente reconocidas. Adicionalmente, no se realizó un análisis con tráfico en tiempo real.

En (Muraleedharan & Janet, 2021) se define un modelo para clasificar ataques DoS tipo Slow con técnicas de aprendizaje profundo, usando el dataset CICIDS2017. Se estableció que la propuesta de los autores está definida en 5 etapas (ver tabla 3-3).

Tabla 3-3: Modelo Aprendizaje Profundo

Etapa 1: Flujo de datos de red	Etapa 2: Preprocesamiento	Etapa 3: Entrenamiento del modelo	Etapa 3: Prueba del modelo	Etapa 5: Validación del modelo
Secuencia de paquetes que se definen entre equipos origen y equipos destinos.	Reemplazo de valores de salida por claves numéricas. Eliminación de 4 columnas para eliminar sesgos (FlowID, timestamp, source and destination IP)	Registros para entrenamiento: 19314 Neuronas en capa de entrada: 80 Función de activación: ReLU	Registros para entrenamiento: 6438	Registros para validación: 6438

Fuente: (Muraleedharan & Janet, 2021).

Como resultados los autores exponen que alcanzó un 100% en términos de precisión, exhaustividad y puntaje F1 para los ataques DoSHulk, DoSGoldenEye y DoS. Se resalta la necesidad de evaluar la precisión del modelo (91%) usando tráfico real.

Fortalezas: Evaluación de métricas en términos de clasificación de ataques DoS. Adicionalmente, se resalta la claridad con respecto a los hiperparámetros de la Red Neuronal Profunda.

Debilidades: Los autores no realizaron una validación de su propuesta con tráfico real ni con usuarios.

Tal como se observó anteriormente, en la comunidad científica se evidencia un interés con respecto a la aplicación de técnicas de inteligencia artificial para la detección de ataques de denegación de servicios distribuidos, y en particular el uso de aprendizaje supervisado, no obstante, quedan abiertos espacios para respuestas explícitas ante los retos de una implementación de un IDS soportado en aprendizaje supervisado.

Por consiguiente, esta propuesta busca la implementación de un sistema de detección de intrusos para la detección de ataques de denegación de servicio distribuido con una arquitectura orientada a servicios, una interfaz gráfica amigable, y con generación de gráficas y reportes, de tal modo que les permita a los administradores de redes integrar o replicar las propuestas de la comunidad científica y tener insumos para la toma decisiones.

A continuación, en la tabla 3-4 se definen algunos parámetros para la construcción de un IDS encontrados en la literatura.

Tabla 3-4: Parámetros para IDS encontrados en literatura

Trabajo	Explicación sobre la propuesta	Propuesta SOA	Notificación o alerta en caso de ataque	Validación con dataset CICIDS2017	Tráfico en tiempo real	Validación con usuarios
(Doriguzzi-Corin et al., 2020)	x					
(Bhardwaj et al., 2020)	x			x		
(Carlin et al., 2015)	x	x				
(Haider et al., 2020)	x			x		
(Puder et al., 2022)	x	x				
(Čelesová et al., 2019)	x				x	
(Virupakshar et al., 2020)	x		x		x	
(T. H. Lee et al., 2020	x					
(Roopak et al., 2020)	x			x		
(Moukhaf et al., 2019)	x					
(Bebortta & Singh, 2021)	x	x				
Taheri et al., 2018)	x				x	x
(Spiekermann & Keller, 2021)	x					
(Shi et al., 2019)	x		x	x	x	
(Alzubi et al., 2022)	x	x				
Dong et al., 2022	x					
(Gangula et al., 2022)	x					
Muraleedharan & Janet, 2021	x			x		

Fuente: Elaboración propia

3.3 Conclusiones del capítulo

En este capítulo, se observa un interés actual en la comunidad científica para detectar ataques DDoS y DoS utilizando diferentes técnicas de aprendizaje supervisado. Sin embargo, la mayoría de las investigaciones se centran en las métricas de predicción y no en proporcionar un mecanismo para los administradores de seguridad en redes mediante servicios en la nube. Los investigadores destacan la necesidad de realizar pruebas con tráfico en tiempo real para evaluar el rendimiento de sus propuestas.

La notificación inmediata por parte de los sistemas de detección de intrusiones (IDS) al analizar el tráfico de red en tiempo real puede proporcionar a los expertos en seguridad informática la información necesaria para tomar medidas correctivas. Además, utilizar sistemas de detección de intrusos basados en arquitecturas orientadas a servicios (SOA) para definir servicios independientes puede agregar valor a las propuestas tradicionales de seguridad informática. Sin embargo, no se han identificado investigaciones que utilicen SOA y aprendizaje supervisado para analizar el tráfico en tiempo real y se hayan realizado pruebas con usuarios y tráfico en tiempo real.

4 Capítulo 4: Técnicas de Aprendizaje Supervisado en Detección de Ataques DDoS

En este capítulo se presenta en detalle las actividades que se desarrollaron en el marco de esta Tesis para determinar cuáles son las técnicas de aprendizaje supervisado a implementar en la construcción del prototipo de IDS. En primera instancia se describe cómo se realizó la preparación de los datos, incluyendo un análisis sobre la posibilidad de reducir el número de atributos del *dataset*. En segunda instancia, se definen las técnicas aprendizaje supervisado analizadas en una investigación conjunta. En última instancia, se presentan las conclusiones del capítulo.

4.1 Preparación de los Datos

En el análisis de datos una de las etapas más importantes es la preparación de los datos. El principal objetivo es garantizar que los datos que serán analizados son de calidad, pueden ser manipulados con facilidad y no contienen ningún tipo de ruido (Croft et al., 2022). A continuación, se describen los ajustes realizados durante el proceso de investigación de esta Tesis hasta tener los datos adecuados para su posterior análisis.

- **Limpieza de los Datos**

  Este proceso consiste en realizar una depuración en los datos con el objetivo de eliminar sesgos, ruido o inconsistencias en los registros a utilizar. El conjunto de datos CICIDS2017, no contiene valores faltantes o registros considerados atípicos, no obstante, se detectaron 79 instancias con caracteres alfanuméricos (*Not at Number – NaN*), los cuales fueron generados a partir de la normalización implementada sobre los datos. Estos registros fueron removidos dado que no afectan considerablemente el número de instancias del *dataset*.

- **Transformación de los Datos**

El proceso de transformación de los datos consistió en realizar actividades que permitieran la manipulación de los datos de una forma práctica, teniendo en cuenta que las técnicas de clasificación requieren que los valores que serán analizados estén en formato numérico. En primera instancia, se procedió con el renombramiento de las columnas, dado que al momento de la descarga del *dataset* se generaron espacios en blanco al inicio del nombre de cada una de las columnas. Se etiquetaron con números enteros, de 0 a 79. En segunda instancia, se generó una codificación con los nombres de cada una de las salidas, de modo tal que cada una de las clases del *dataset* sea identificada con un número entero. Por último, para que los datos estén comprendidos en una misma escala se generó una normalización de los valores por medio de la librería Sklearn.

- **Eliminación de Datos**

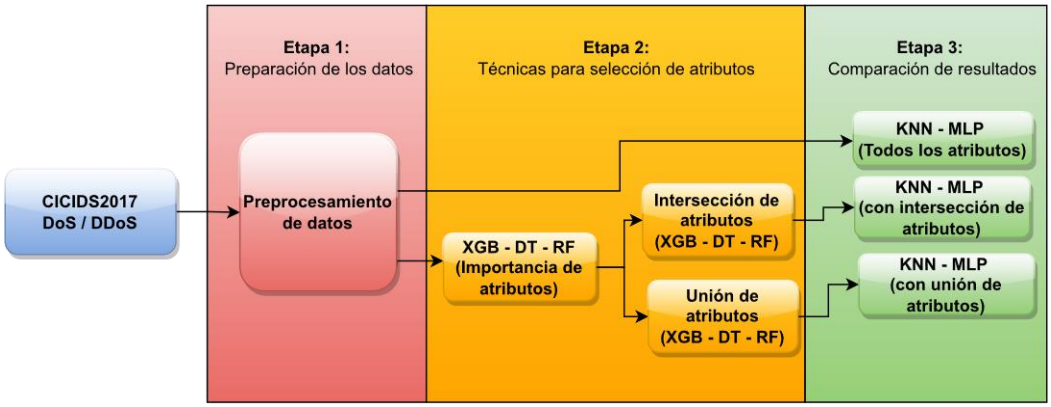
  La remoción de valores en un conjunto de datos es utilizada cuando no se consideran relevantes para su análisis, en el caso del dataset CICIDS2017, se encontró una columna repetida (Fwd Header Length.1), por lo cual fue no fue tenida en cuenta. Sumado a lo anterior, dado el alcance de esta Tesis, fueron excluidos los datos relacionados con otros tipos de ataques diferentes a DoS/DDoS. A continuación, se definen las categorías de ataques seleccionadas.

  - ✓ BENIGN
  - ✓ DDoS
  - ✓ DoS GoldenEye
  - ✓ DoS Hulk
  - ✓ DoS Slowhttptest
  - ✓ DoS slowloris

4.2 Selección de Características

La selección de características es una estrategia implementada en el análisis de datos que busca una reducción en la dimensionalidad de un problema, y en particular, dada la cantidad de *columnas* del dataset CICIDS2017 (79) se tomó la decisión de evaluar la viabilidad de encontrar un subconjunto de atributos que permitan clasificar adecuadamente un ataque de tipo DoS/DDoS. Para la aplicación de la selección de atributos se definió una metodología por etapas, tal como se observa en la figura 4-1.

Figura 4-1: Metodología por Etapas para Selección de Atributos



Fuente: Elaboración propia

Tal como se observa en la figura 4-1, la metodología propuesta consta de 3 etapas: preprocesamiento de los datos, técnicas para la selección de atributos y comparación de los resultados. El porcentaje de importancia de cada uno de los atributos del *dataset*, se determinó por medio de la librería Sklearn y su atributo “feature_importances_”, tal como se realizó en (J. Wang et al., 2021), (Leevy et al., 2021) y (Sharma & Yadav, 2021). Sin embargo, éste atributo requiere de una técnica de aprendizaje supervisado para otorgar la importancia de cada característica, por lo cual se definieron como algoritmos de clasificación a XGB, DT y RF. Se decidió que serían considerados los atributos hasta que la sumatoria acumulada alcanzara el 80%.

A continuación, se definen los porcentajes de importancia de cada uno de los atributos según XGB, DT y RF, tal como se observa en las tablas 4-1, 4-2 y 4-3.

Tabla 4-1: Importancia de atributos según DT

Atributo	% de importancia	% de Importancia (acumulado)
75	22,50997%	22,50997%
42	17,87567%	40,38564%
0	13,88353%	54,26917%
36	13,20165%	67,47083%
71	8,34455%	75,81537%
6	4,38788%	80,20326%

***Fuente*:** Elaboración propia

Tabla 4-2: Importancia de atributos según XGB

Atributo	% de importancia	% de Importancia (acumulado)
3	16,00%	16%
35	8,67%	25%
36	6,42%	31%
38	6,35%	37%
4	6,16%	44%
25	5,26%	49%
0	3,85%	53%
7	3,29%	56%
42	3,20%	59%
6	2,93%	62%
67	2,89%	65%
75	2,79%	68%
11	2,48%	70%
22	2,27%	73%
24	2,18%	75%
65	2,05%	77%
14	1,78%	79%
23	1,76%	80%

***Fuente*:** Elaboración propia

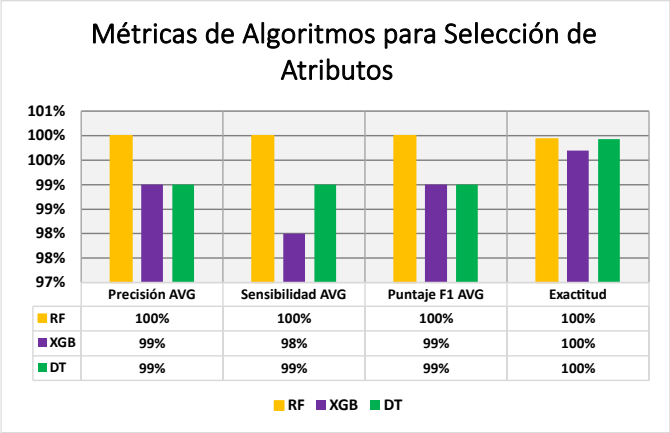
Tabla 4-3: Importancia de atributos según RF

Atributo	% de importancia	% de importancia (acumulado)
36	5,366%	5,366%
75	4,782%	10,148%
73	4,638%	14,786%
76	4,436%	19,222%
0	4,382%	23,604%
1	4,138%	27,742%
22	3,864%	31,606%
42	3,615%	35,221%
65	3,533%	38,754%
15	3,461%	42,215%
23	3,299%	45,514%
4	3,064%	48,578%
13	2,761%	51,339%
14	2,584%	53,923%
21	2,352%	56,275%
20	2,220%	58,495%
62	1,957%	60,452%
17	1,779%	62,231%
16	1,758%	63,989%
18	1,747%	65,735%
34	1,727%	67,462%
69	1,650%	69,112%
5	1,632%	70,744%
71	1,611%	72,355%
6	1,584%	73,940%
10	1,479%	75,419%
72	1,401%	76,819%
27	1,389%	78,208%
25	1,373%	79,581%
9	1,363%	80,943%

***Fuente*:** Elaboración propia

A continuación en la figura 4-2 se detallan las métricas obtenidas por parte de los clasificadores XGB, RF y DT.

Figura 4-2: Métricas de clasificadores para selección de características



***Fuente*:** Elaboración propia

A partir de los atributos definidos en las tablas 4-1, 4-2 y 4-3, se definieron 3 subconjuntos de atributos: El completo, y los obtenidos con la unión y la intersección, tal como se observa en la tabla 4-4 y 4-5 respectivamente.

Tabla 4-4: Subconjunto Intersección

Subconjunto	Atributos
Intersección	0, 36, 6, 42, 75

***Fuente*:** Elaboración propia

Tabla 4-5: Subconjunto Unión

Subconjunto	Atributos
Unión	0, 1, 3, 4, 5, 6, 7, 9, 10, 11, 13, 14, 15, 16, 17, 18, 20, 21, 22, 23, 24, 25, 27, 34, 35, 36, 38, 42, 62, 65, 67, 69, 71, 72, 73, 75, 76

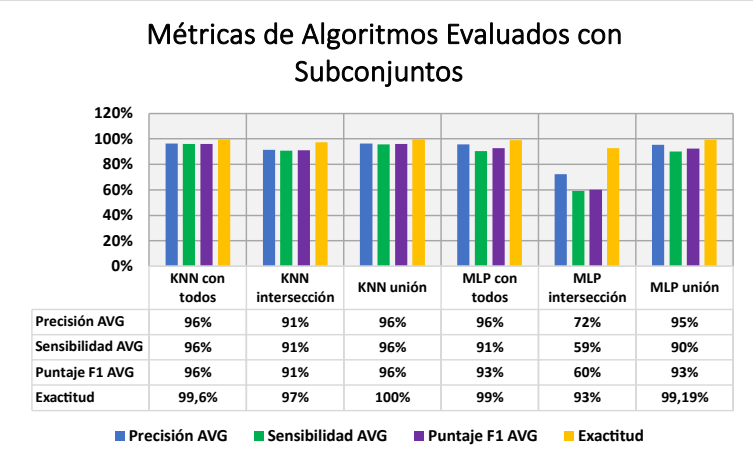
***Fuente*:** Elaboración propia

Los subconjuntos definidos en las tablas 4-4 y 4-5 fueron usados como valores de entrada para validar si era posible un buen rendimiento en términos de clasificación con estos atributos. Las técnicas de aprendizaje supervisado usadas para la validación fueron KNN y MLP. De allí se generaron 3 mediciones:

- ✓ KNN y MLP usando todos los atributos (77 columnas).
- ✓ KNN y MLP usando los 5 atributos definidos en la intersección.
- ✓ KNN y MLP usando los 37 atributos definidos en la unión.

A partir de lo mencionado anteriormente, los clasificadores obtuvieron las métricas que se exponen en la figura 4-3.

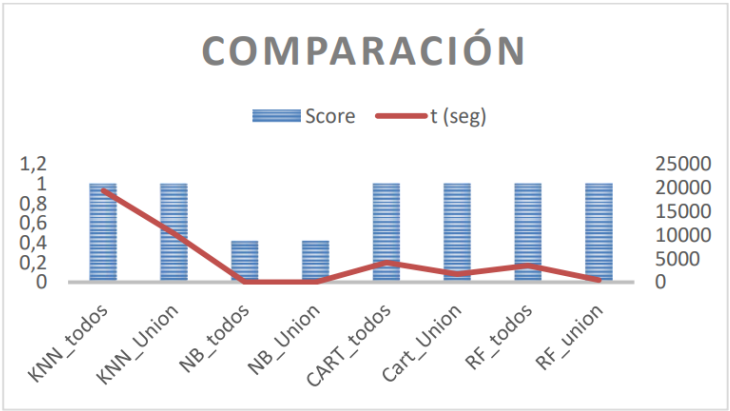
Figura 4-3: Métricas de algoritmos evaluados con subconjuntos de atributos



***Fuente*:** Elaboración propia

Adicionalmente, se realizaron cálculos para determinar el rendimiento de la clasificación con el subconjunto de atributos definidos en la unión (Montes-Gil et al., 2023).

Figura 4-4: Comparación de Resultados entre Diferentes Técnicas



***Fuente*:** (Montes-Gil et al., 2023)

Tal como se puede observar en la figura 4-3, el rendimiento de los clasificadores con los datos de prueba se ve comprometido en los casos donde se usó el subconjunto de atributos con intersección y hay una leve reducción con el subconjunto obtenido con la unión (37 atributos), es decir, a medida que se disminuye el número de atributos. De allí se tomó la decisión, en esta investigación, de aplicar las técnicas para modelos de clasificación con todo el conjunto de atributos. Sin embargo, en la figura 4-4 se observan resultados promisorios que deben ser analizados con mayor detenimiento para concluir la viabilidad de la reducción de atributos en el dataset CICIDS2017.

4.3 Definición de Modelos Basados en Técnicas de Aprendizaje Supervisado

Para la definición de las técnicas de aprendizaje supervisado se parte de una Tesis de Maestría en desarrollo titulada "*Prototipo de Red Neuronal Profunda Aplicada en Ciberseguridad*" (Isaza & Ramírez, 2023), la cual es desarrollada en paralelo a esta investigación y se encuentra inmersa en el proyecto "Prototipo para Detección de ataques de DDoS (Denegación de Servicios Distribuida) Basado en Aprendizaje de Máquina en una Arquitectura Orientada a Servicios en la Nube", la cual realiza un análisis al conjunto de datos CICIDS2017 con técnicas de aprendizaje supervisado, se determinan que las técnicas de aprendizaje supervisado con mejores métricas en términos de clasificación son:

- **KNN.**
- **Red Neuronal Profunda.**
- **Red Neuronal Convolucional.**

A continuación, se describe en detalle el proceso de aumento de datos generado para identificar firmas de ataques actuales, las características de los modelos propuestos y los resultados en fase de entrenamiento y prueba.

4.3.1 Aumento de Datos

Teniendo en cuenta las etiquetas en el dataset CICID2017 (ver tabla 4-6), es evidente que se presenta un desbalanceo en los datos, con un alto número de registros benignos y unos pocos, en particular para los ataques DoSSlowloris, DoSSlowhttptest y DoSGoldeneye.

De allí surge la necesidad de aplicar alguna estrategia para balancear el dataset. Además de la necesidad de aumentar el número de registros, se evidenció la necesidad de entrenar los modelos con tramas reales generadas por los ataques en ambientes controlados, lo cual permitirá agregar registros de ataques que fueron capturados después de la creación del dataset.

Tabla 4-6: Total de instancias por cada clase

Clase	Registros iniciales	Registros finales
Benigno	2273097	2287872
DDoS	128027	271867
DoS Goldeneye	10293	40377
DoS Hulk	231073	288959
DoS Slowhttptest	5499	25781
DoS Slowloris	5796	50476

Fuente: Elaboración propia

Los ataques que se lanzaron en ambientes controlados para obtener un aumento de datos fueron ejecutados por las herramientas definidas en la tabla 4-7.

Tabla 4-7: Herramientas usadas para aumento de datos (ataques)

Herramienta	Enlace
Low Orbit Ion Cannon	https://sourceforge.net/projects/loic/
High Orbit Ion Cannon	https://sourceforge.net/projects/highorbitioncannon
GoldenEye	https://github.com/jseidl/GoldenEye
Hulk	https://github.com/grafov/hulk.git
Slowloris	https://github.com/Dafa2019/slowloris
Slowhttptest	https://github.com/shekyan/slowhttptest

Fuente: Elaboración propia

Proceso de Captura de Tráfico en Ataques

Inicialmente los ataques eran lanzados y capturados por medio de la herramienta Wireshark y TCPDump sin realizar ningún tipo de filtro, no obstante, se observaron clasificaciones erróneas por parte de los modelos en fases de entrenamiento. Esto se debía a que los ataques pueden intentar emular el comportamiento de una trama benigna, por medio de protocolos como ICMP, SMB, ARP, entre otros. A continuación en la tabla 4-8 se detallan las reglas definidas en Wireshark para obtener tráfico de ataques sin características de tramas benignas.

Tabla 4-8: Reglas definidas en Wireshark

Ataque	Regla
DoSSlowloris	not icmpv6 && not ipv6 && not ssdp && not nbdgm && not smb && not mailslot && not browser && not mdns && not dns && not icmp && not arp
DDoS (herramienta HOIC)	not ipv6 && not icmpv6 && not nbdgm && not smb && not mailslot && not browser && not mdns && not dhcp && not dns && not arp
DoSSlowhttptest	not ssdp && not nbdgm && not smb && not mailslot && not browser && not mdns && not dns && not icmp && not arp
DoSGoldeneye	not icmpv6 && not ipv6 && not ssdp && not nbdgm && not smb && not mailslot && not browser && not mdns && not dns && not icmp && not arp
DoSHulk	not arp && not dns && not icmp

Fuente: (Isaza & Ramírez, 2023)

Las reglas definidas en la table 4-8 fueron implementadas durante el lanzamiento de los ataques para obtener un mejor proceso de entrenamiento con los datos que se incluyeron.

K Vecinos Cercanos

Para el clasificador KNN se definieron 10 vecinos como uno de los hiperparámetros, así como el estado aleatorio (random_state) en 42.

A continuación, en la tabla 4-9 se muestran las métricas definidas en fase de entrenamiento, mientras que en la tabla 4-10 se definen los resultados para la fase de validación.

Tabla 4-9: Reporte de clasificación en fase de entrenamiento para KNN

Clase	Precisión (*precision*)	Sensibilidad (*recall*)	Puntaje f1 (f1 score)	Instancias
Benigno	0.9977	0.9965	0.9971	1842581
DDoS	0.9959	0.9950	0.9955	332481
DoS Goldeneye	0.9620	0.9889	0.9753	56442
DoS Hulk	0.9863	0.9872	0.9867	278009
DoS Slowhttptest	0.7959	0.8452	0.8198	36782
DoS Slowloris	0.8966	0.8787	0.8876	76204
Exactitud (acurracy)			0.9896	2622589
(Macro avg)	0.9391	0.9486	0.9437	2622589
(Weighte avg)	0.9898	0.9896	0.9897	2622589

Fuente: (Isaza & Ramírez, 2023)

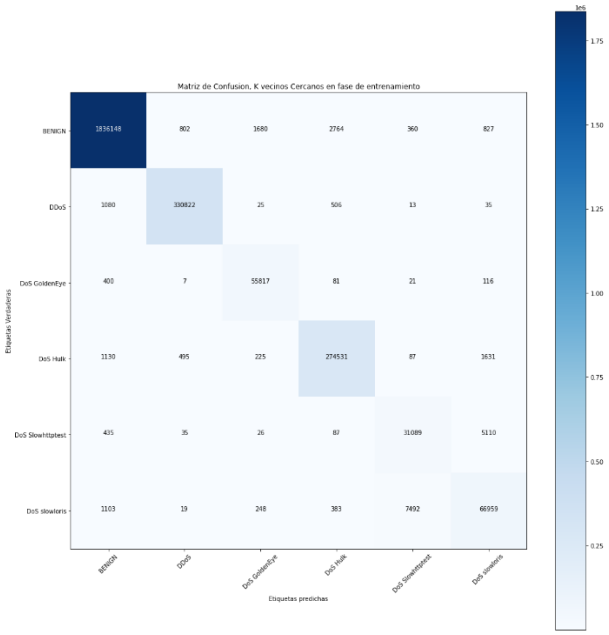
Tabla 4-10: Reporte de clasificación en fase de validación para KNN

Clase	Precisión (*precision*)	Sensibilidad (*recall*)	Puntaje f1 (f1 score)	Instancias
Benigno	0.9975	0.9961	0.9968	460475
DDoS	0.9949	0.9952	0.9951	83226
DoS Goldeneye	0.9584	0.9884	0.9731	14019
DoS Hulk	0.9845	0.9842	0.9844	69695
DoS Slowhttptest	0.8026	0.8457	0.8236	9281
DoS Slowloris	0.98913	0.8776	0.8844	18952
Exactitud (acurracy)			0.9890	655648
(Macro avg)	0.9382	0.9479	0.9429	655648
(Weighte avg)	0.9891	0.9890	0.9891	655648

Fuente: (Isaza & Ramírez, 2023)

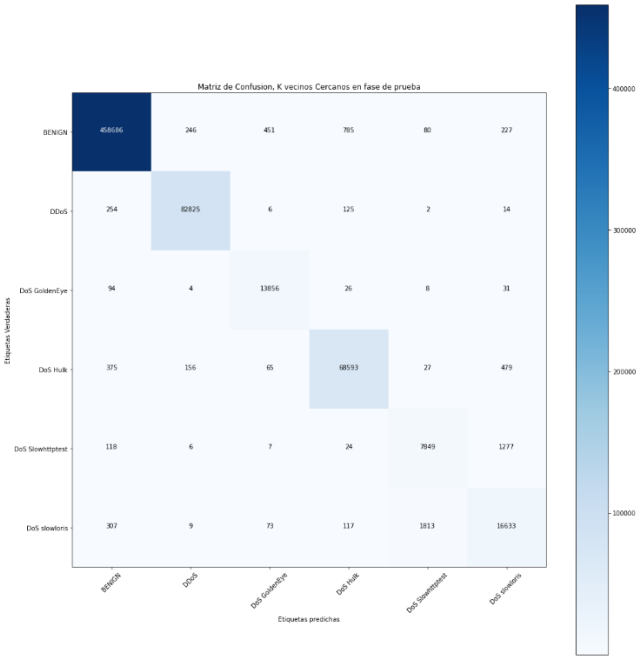
La figura 4-5 muestra la matriz de confusión para el clasificador KNN en fase de entrenamiento, mientras que la figura 4-6 define la matriz de confusión para la fase de prueba.

Figura 4-5: Matriz de Confusión KNN fase de entrenamiento



Fuente: (Isaza & Ramírez, 2023)

Figura 4-6: Matriz de Confusión KNN fase de prueba



Fuente: (Isaza & Ramírez, 2023)

Red Neuronal Profunda

En la tabla 4-11 se definen los elementos que conforman la configuración de la red neuronal profunda.

Tabla 4-11: Configuración Red Neuronal Profunda

Elementos de la Red Neuronal Profunda			
Capas ocultas	1	2	3
Neuronas	130	83	78
Función activación	Relu	Relu	Softmax
Tamaño de lote		200	
Tasa de aprendizaje		0.001	
división de validación		0.2	
Número de épocas		13	
Beta 1		0.9	
Beta 2		0.999	
Elipsón		$1x10^{-7}$	

Fuente: (Isaza & Ramírez, 2023)

Las tablas 4-12 y 4-13 detallan las métricas en fase de entrenamiento y validación.

Tabla 4-12: Reporte de clasificación en fase de entrenamiento para Red Neuronal Profunda

Clase	Precisión (*precision*)	Sensibilidad (*recall*)	Puntaje f1 (f1 score)	Instancias
Benigno	0.9970	0.9949	0.9960	1830889
DDoS	0.9895	0.9923	0.9909	217046
DoS Goldeneye	0.9465	0.9703	0.9583	32319
DoS Hulk	0.9702	0.9833	0.9767	231026
DoS Slowhttptest	0.7051	0.9095	0.7944	20589
DoS Slowloris	0.9291	0.7741	0.8445	40396
Exactitud (acurracy)			0.9887	2372265
(Macro avg)	0.9229	0.9374	0.9268	2372265
(Weighte avg)	0.9893	0.9887	0.9888	2372265

Fuente: (Isaza & Ramírez, 2023)

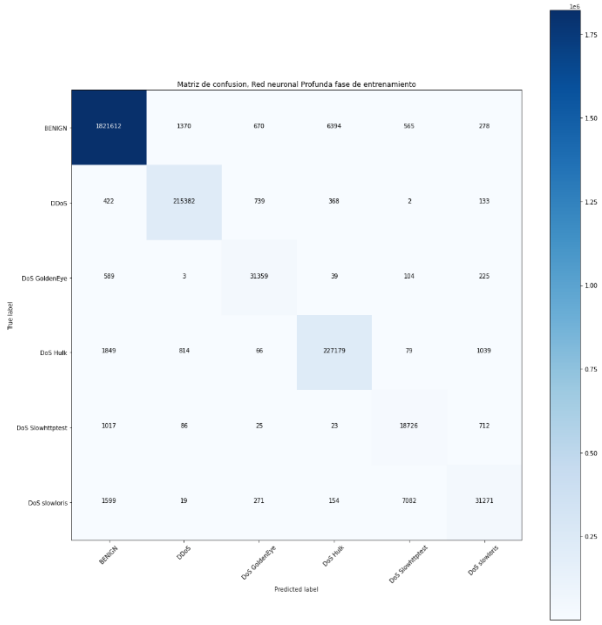
Tabla 4-13: Reporte de clasificación en fase de validación para Red Neuronal Profunda

Clase	Precisión (*precision*)	Sensibilidad (*recall*)	Puntaje f1 (f1 score)	Instancias
Benigno	0.9969	0.9950	0.9960	456983
DDoS	0.9906	0.9925	0.9916	54821
DoS Goldeneye	0.9500	0.9710	0.9604	8058
DoS Hulk	0.9701	0.9834	0.9767	57933
DoS Slowhttptest	0.7039	0.9039	0.7915	5192
DoS Slowloris	0.9251	0.7695	0.8402	10080
Exactitud (acurracy)			0.9887	593067
(Macro avg)	0.9228	0.9359	0.9260	593067
(Weighte avg)	0.9893	0.9887	0.9887	593067

Fuente: (Isaza & Ramírez, 2023)

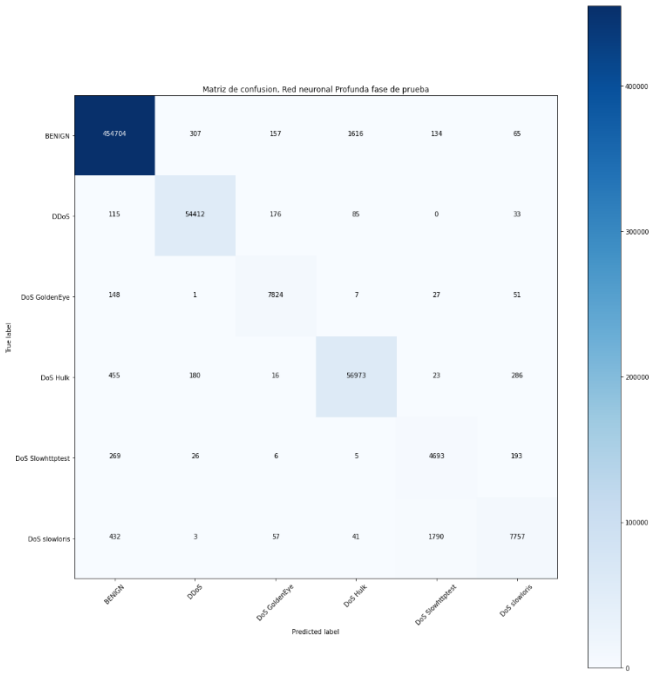
En las figuras 4-7 y 4-8 se observan las matrices de confusión de acuerdo con el reporte de entrenamiento y prueba.

Figura 4-7: Matriz de Confusión DNN fase de entrenamiento



Fuente: (Isaza & Ramírez, 2023)

Figura 4-8: Matriz de Confusión DNN fase de prueba



Fuente: (Isaza & Ramírez, 2023)

Red Neuronal Convolucional

En la tabla 4-14 y 4-15 se relacionan los valores de configuración de la red neuronal convolucional.

Tabla 4-14: Configuración de la red neuronal convolucional

Elementos de la Red Neuronal Convolucional					
Capa 1	Convolucional 2D	Número de neuronas	100	Función de activación	Relu
Capa 2	Max-Pooling		Tamaño	(1,2)	
Capa 3	Flatten (aplanamiento)				
Capa 4	Densa	Número de neuronas	80	Función de activación	Relu
Capa 5	Densa	Número de neuronas	78	Función de activación	Softmax

Fuente: (Isaza & Ramírez, 2023)

Tabla 4-15: Hiperparámetros Red Neuronal Convolucional

Hiperparámetros Red Neuronal Convolucional	
Tamaño de lote	200
Tasa de aprendizaje	0.001
división de validación	0.2

Fuente: (Isaza & Ramírez, 2023)

En las tablas 4-15 y 4-16 se detallan las métricas alcanzadas en las fases de entrenamiento y de prueba.

Tabla 4-16: Métricas para red neuronal convolucional en fase de entrenamiento

Clase	**Precisión (*precision*)**	**Sensibilidad (*recall*)**	**Puntaje f1 (f1 score)**	**Instancias**
Benigno	0.9962	0.9896	0.9929	1830889
DDoS	0.9913	0.9885	0.9899	217046
DoS Goldeneye	0.7283	0.9774	0.8347	32319
DoS Hulk	0.7283	0.9774	0.8347	231026
DoS Slowhttptest	0.9636	0.9835	0.9734	20589
DoS Slowloris	0.7212	0.8555	0.7826	40396
Exactitud (acurracy)			0.9840	2372265
(Macro avg)	0.8904	0.9292	0.9045	2372265
(Weighte avg)	0.9856	0.9840	0.9843	2372265

Fuente: (Isaza & Ramírez, 2023)

Tabla 4-17: Métricas para red neuronal convolucional en fase de prueba

Clase	Precisión (*precision*)	Sensibilidad (*recall*)	Puntaje f1 (f1 score)	Instancias
Benigno	0.9959	0.9896	0.9928	456983
DDoS	0.9919	0.9881	0.9900	54821
DoS Goldeneye	0.7259	0.9769	0.8329	8058
DoS Hulk	0.9641	0.9827	0.9733	57933
DoS Slowhttptest	0.7239	0.8478	0.7810	5192
DoS Slowloris	0.9379	0.7805	0.8520	10000
Exactitud (acurracy)			0.9838	593067
(Macro avg)	0.8899	0.9276	0.9838	593067
(Weighte avg)	0.9854	0.9838	0.9842	593067

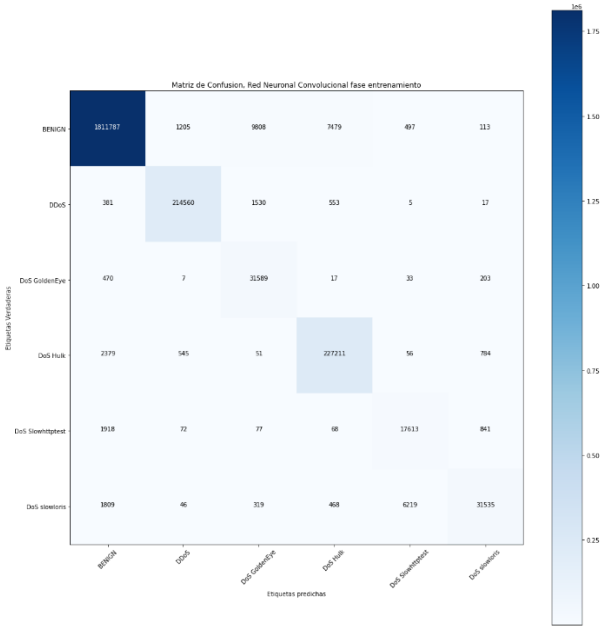
Fuente: (Isaza & Ramírez, 2023)

Número de épocas	20
Beta 1	0.9
Beta 2	0.999
Elipsón	$1x10^{-7}$

Fuente: (Isaza & Ramírez, 2023)

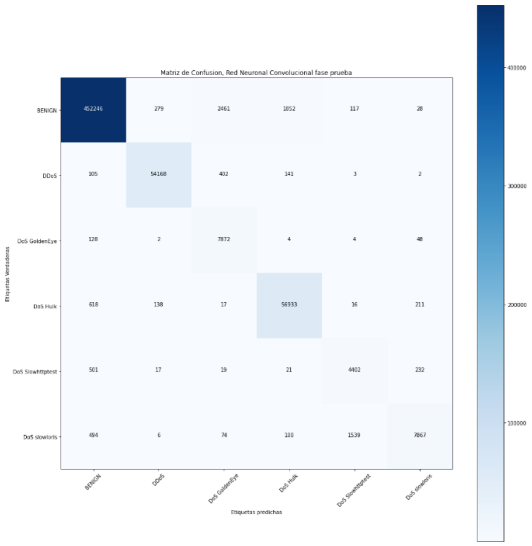
A continuación, en la figura 4-9 se observa la matriz de confusión para la red convolucional en fase entrenamiento, y para la fase de prueba en la figura 4-10.

Figura 4-9: Matriz de confusión CNN fase de entrenamiento



Fuente: (Isaza & Ramírez, 2023)

Figura 4-10: Matriz de confusión CNN fase de prueba



Fuente: (Isaza & Ramírez, 2023)

Los resultados en los modelos definidos en (Isaza & Ramírez, 2023), KNN, DNN y CNN, son promisorios en términos de clasificación, tanto en la etapa de entrenamiento como en la fase de prueba, por lo tanto, en esta Tesis serán usados para la construcción de un prototipo de IDS soportado en técnicas de aprendizaje supervisado para la detección de ataques de tipo DoS/DDoS con un enfoque de servicios en la nube.

4.4 Conclusiones del Capítulo

En este capítulo se definieron las actividades realizadas desde la preparación de los datos hasta la determinación de las técnicas de aprendizaje supervisado a implementar en el prototipo de IDS. Luego de llevar a cabo un análisis de selección de atributos sobre el conjunto de datos, se determinó que reducir el número de atributos no es una opción relevante para mejorar las clasificaciones generadas por las técnicas de aprendizaje supervisado. De hecho, se observó que las clasificaciones disminuyen en su eficacia a medida que se reduce el número de atributos. Por lo tanto, se decidió no aplicar una reducción de características en el curso de la investigación. Se llevó a cabo un análisis de las técnicas de aprendizaje supervisado disponibles y se concluyó que las técnicas más apropiadas para implementar en el prototipo de detección de intrusos son K-Nearest Neighbors (KNN), Convolutional Neural Network (CNN) y Deep Neural Network (DNN), debido a su buen rendimiento demostrado en las fases de entrenamiento y prueba. Además, se constató que la ampliación de los datos generados a través de la inyección de

ataques tuvo un impacto significativo en la mejora de la clasificación y en la obtención de resultados concluyentes en la etapa de validación.

5 Capítulo 5: Diseño de la Arquitectura del Prototipo Orientado a Servicios e Implementación Basado en Aprendizaje Supervisado en Arquitectura Orientada a Servicios en la Nube

En este capítulo se presenta el diseño de la arquitectura del prototipo, tanto en su orientación a la nube como en su funcionamiento a nivel local, detallando cada uno de los servicios que se ofrecen y su funcionamiento. Además de lo anterior, se describe el proceso de análisis, diseño e implementación, haciendo énfasis en las tecnologías definidas en los ambientes de desarrollo, pruebas y de producción.

5.1 Diseño de la Arquitectura del Prototipo Orientado a Servicios en la Nube

Etapa 1 – Envío de archivo .pcap: En esta etapa el usuario realiza el envío de la captura de tráfico realizada por medio de un servicio en la nube. Este servicio se define en un aplicativo web, en el cual se define la autenticación, registro de usuarios y reportes.

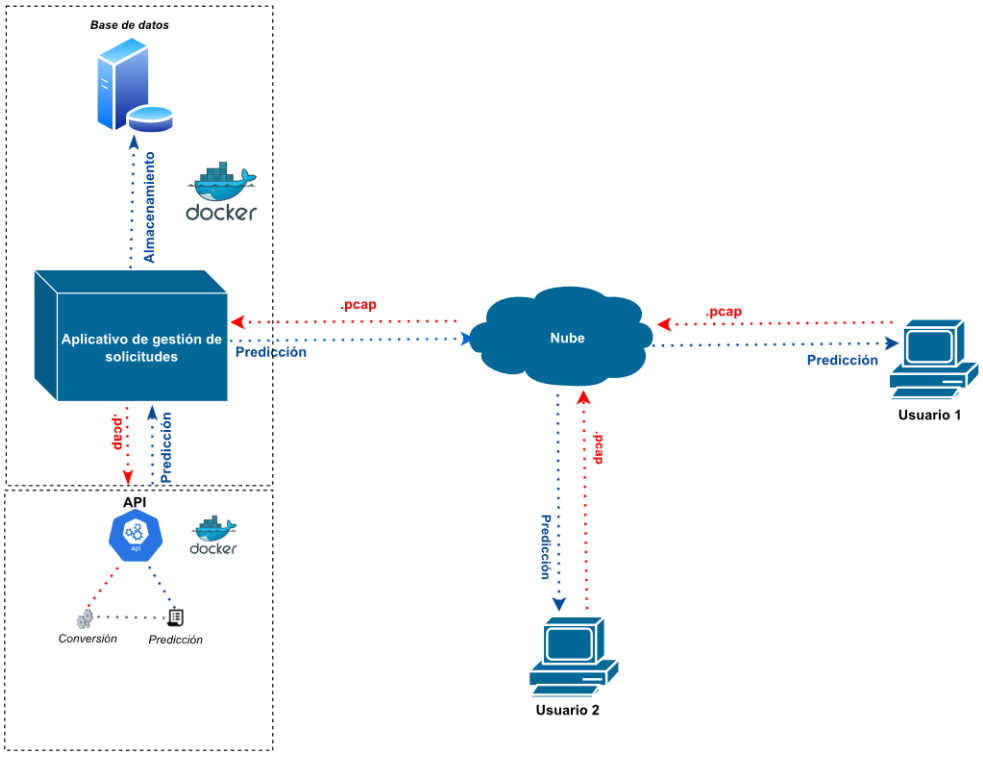
Etapa 2 - Conversión a formato CICIDS2017: La conversión al formato del dataset CICIDS2017 se realiza de manera automática por medio de una API que recibe el archivo .pcap enviado en la etapa 1.

Etapa 3 – Técnicas de Aprendizaje Supervisado: Aquí los modelos de aprendizaje supervisado generan la predicción de manera automático, retornando al aplicativo web los resultados de la clasificación.

Etapa 4 – Reportes: Por último, luego de recibir los resultados generados en la etapa 3, el aplicativo almacena las predicciones en una base de datos para garantizar la persistencia de los datos.

A continuación, en la figura 5-1 se define la arquitectura del prototipo orientado a servicios en la nube.

Figura 5-1: Arquitectura del Prototipo Orientado a Servicios en la Nube



Fuente: Elaboración propia

Tal como se puede observar en la figura 5-1, el servicio desplegado en la nube permite realizar predicciones desde diferentes equipos dada su disponibilidad como un servicio. Adicionalmente, soporta más de una petición al mismo tiempo, lo cual garantiza una concurrencia de usuarios.

A continuación, se definen los detalles de la implementación del servicio en la nube.

5.2 Implementación del Prototipo Basado en Aprendizaje Supervisado en Arquitectura Orientada a Servicios en la Nube

5.2.1 Listado de Requerimientos

Análisis de Requerimientos

El análisis de requerimientos es la etapa en la cual se identifican las necesidades que deben ser solucionadas por medio del sistema a implementar. Una de las estrategias más usadas en los diferentes procesos de ingeniería de software, es la categorización de cada necesidad en términos de una solución; a este proceso se le conoce como identificación de requerimientos.

Los requerimientos pueden ser clasificados en dos categorías, requerimientos funcionales y requerimientos no funcionales.

Requerimientos Funcionales: Representan las actividades que se deben realizar en el sistema.

Requerimientos no Funcionales: Definen las restricciones sobre las cuales será implementado el sistema.

A continuación, en la tabla 5-1 se define el listado de requerimientos funcionales y en la tabla 5-2 los requerimientos no funcionales.

Tabla 5-1: Listado de requerimientos funcionales

#	Requerimiento	Descripción	Prioridad
RF1	Registro de usuarios	El sistema debe permitir al usuario registrarse con nombre, correo electrónico y contraseña.	5
RF2	Autenticación	El sistema debe permitir al usuario iniciar sesión por medio de la contraseña y su correo electrónico.	5
RF3	Envío de archivo en formato .pcap	El sistema debe permitir cargar y enviar un archivo en formato PCAP	5

RF4	Analizar, convertir, clasificar y almacenar automáticamente	El sistema debe realizar la conversión a formato CICIDS2017, definir la predicción y retornar las predicciones para que sean almacenadas en una base de datos.	5
RF5	Generación de reportes	El sistema debe permitir la renderización de gráficas por cada predicción realizada.	5

Fuente: Elaboración propia

Tabla 5-2: Listado de requerimientos no funcionales

#	Requerimiento	Descripción	Prioridad
RNF1	Comportamiento responsive	El prototipo debe estar en la capacidad de adaptarse a diferentes dispositivos.	3
RNF2	Interfaz sencilla e intuitiva	El prototipo debe tener una interfaz clara, sin sobre carga de elementos y buen contraste.	4
RNF3	Menú de navegación	El prototipo debe tener un menú de navegación para acceder fácilmente a las diferentes opciones.	4
RNF4	Validaciones de formularios tanto del lado del cliente como del lado del servidor	Las validaciones de los formularios deben hacerse desde el lado del cliente por medio de HTML5 y JavaScript, al igual que del lado del servidor, para aumentar los niveles de seguridad del prototipo.	5
RFN5	Documentación del código	El sistema debe contar con documentación del código para garantizar facilidad en el mantenimiento y mejoras.	3
RNF6	Incluir prácticas de accesibilidad web y glosario	El prototipo debe tener formularios con características mínimas de accesibilidad (nivel A), al igual que un glosario para asociar términos y definiciones.	4

Fuente: Elaboración propia

5.2.2 Implementación

A continuación, en la tabla 5-3 se resumen las tecnologías usadas en la implementación del servicio en la nube.

Tabla 5-3: Tecnologías usadas

Finalidad	Tecnología
API para conversión y predicción	Flask
Aplicación para gestion de solicitudes	Laravel
Conversión de tráfico a formato CSV	CICFlowmeter
Virtualización de Python	Pyenv: 3.8.5
Reportes (Gráficos)	Plotly JS
Base de datos	MySQL
Tecnología de virtualización	Docker

Fuente: Elaboración propia

En las figuras 5-2 y 5-3 se definen los formularios de registro de usuario e inicio de sesión.

Autenticación y registro: Para poder tener un control sobre la cantidad de peticiones realizadas y almacenar los registros de las predicciones, se definió un mecanismo de registro y autenticación, tal como se observa en las figuras 5-2 y 5-3.

Figura 5-2: Formulario de registro de usuarios

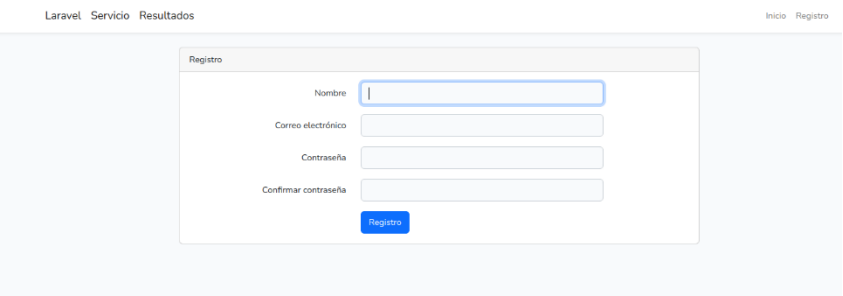


Figura 5-3: Formulario de inicio de sesión

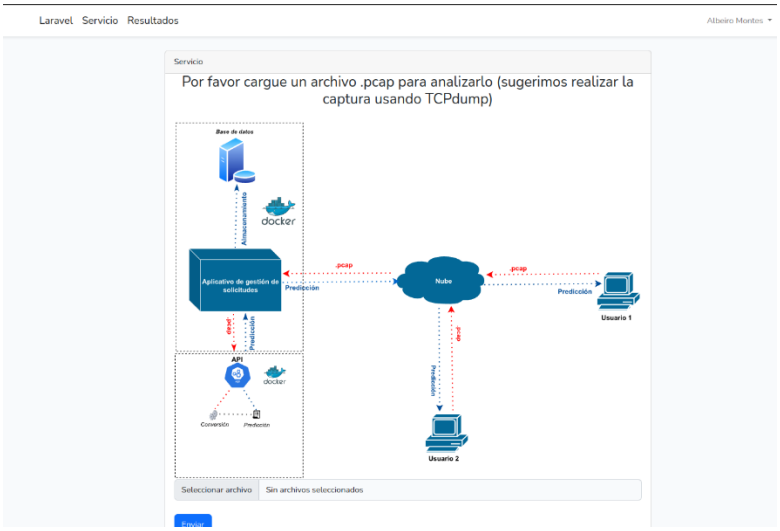
Fuente: Elaboración propia

Envío de archivo .pcap: El envío del archivo en formato .pcap se realiza por medio de un formulario en la aplicación de gestión. Para acceder a este servicio, es necesario realizar un registro con nombre, correo electrónico y contraseña, tal como se observó en las figuras 5-2 y 5-3.

Se definió que el archivo .pcap debe ser generado por el usuario, dada las implicaciones de seguridad que conlleva el obtener el tráfico de red de un equipo por fuera del segmento de red en el cual estará desplegado el servicio en la nube.

Una de las recomendaciones definidas, es la captura del tráfico de red por medio de la herramienta TCPdump, dada su facilidad de instalación, velocidad en la captura, versatilidad para capturar tráfico por medio de diferentes interfaces de red y el buen rendimiento observado en los ambientes de prueba realizados. A continuación, en la figura 5-4 se observa el formulario definido para el envío del archivo .pcap.

Figura 5-4: Carga de archivo .pcap

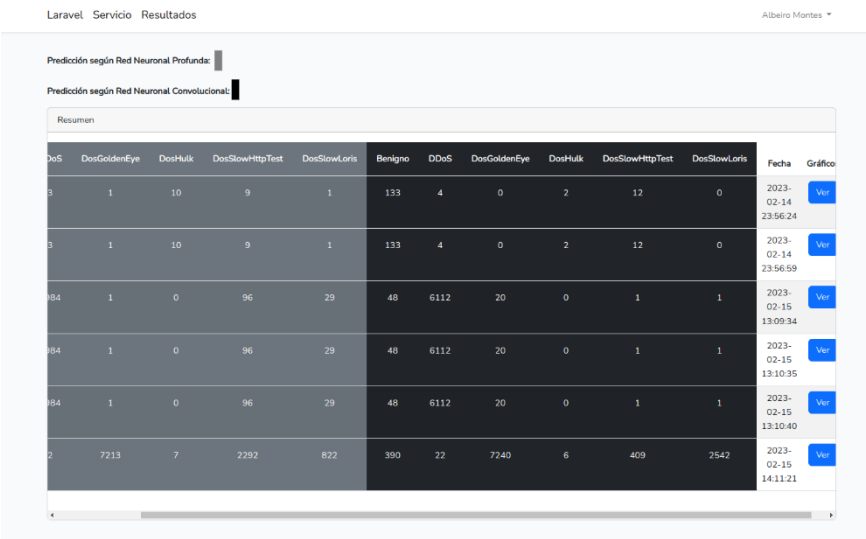


Fuente: Elaboración propia

Luego de que el archivo es enviado a la API de Flask que permite realizar la conversión al formato del dataset CICIDS2017 y obtener las predicciones por medio de los modelos de red neuronal profunda y red neuronal convolucional, los registros son almacenados en la base de datos MySQL y renderizados en una tabla, tal como se observa en la figura 5-5.

Cabe resaltar, que para mejorar los tiempos de respuesta en el servicio en la nube, se decidió no incluir el clasificador KNN.

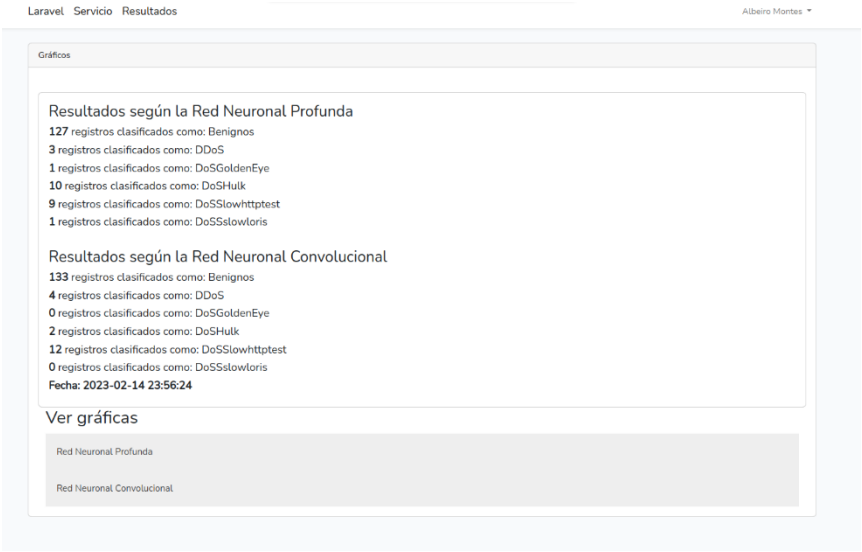
Figura 5-5: Tabla con predicciones generadas



Fuente: Elaboración propia

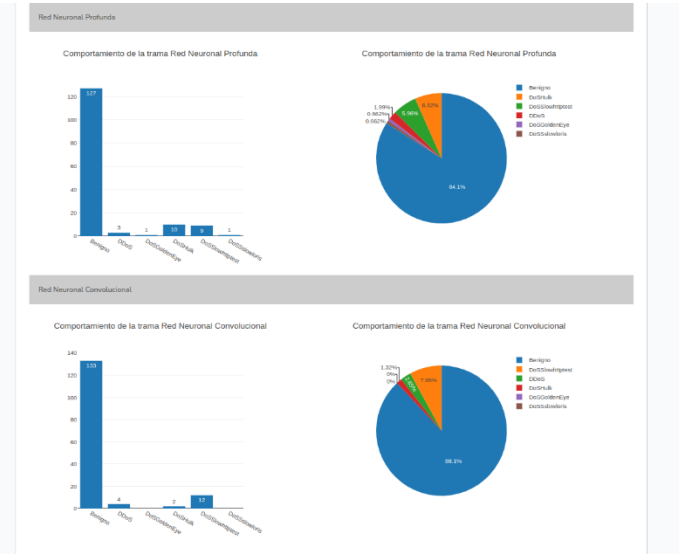
Reportes: Por cada una de las predicciones, se pueden encontrar gráficos que representan los resultados según el tráfico enviado por el usuario, tal como se observa en la figura 5-6 y 5-7.

Figura 5-6: Resultados por cada modelo de predicción



Fuente: Elaboración propia

Figura 5-7: Gráficos por cada modelo de predicción



Fuente: Elaboración propia

5.3 Diseño de la Arquitectura del Prototipo Orientado a Servicios

El prototipo desarrollado en una arquitectura orientada a servicios que permite realizar capturas de tráfico, conversión de datos y clasificación con módulo de reportes, notificación y gráficas. A diferencia del Prototipo desplegado en la nube, este aplicativo puede ser instalado a nivel local por el usuario. A continuación, se resumen cada una de las etapas definidas en la arquitectura del sistema.

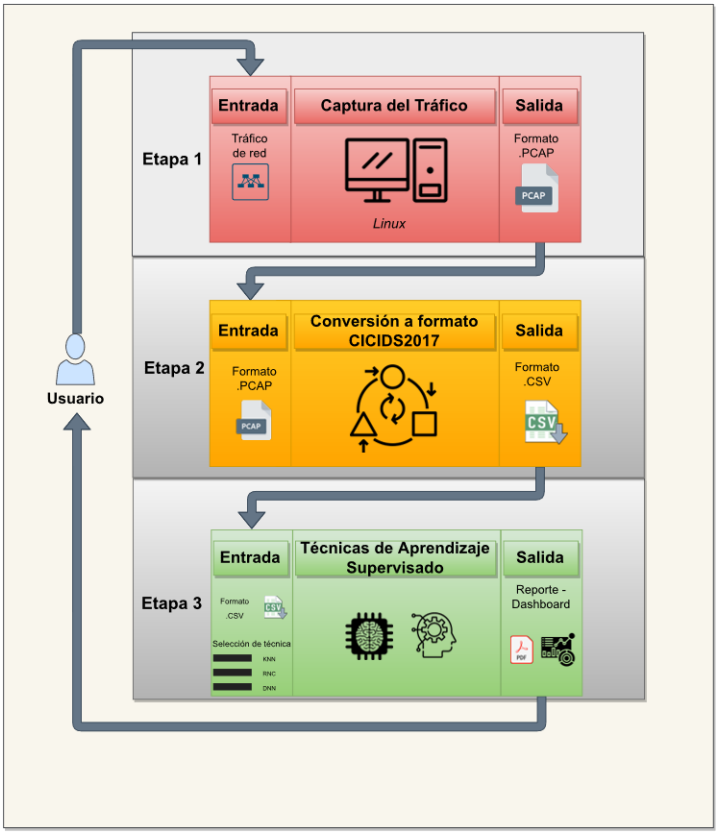
Etapa 1 – Captura de tráfico: En esta etapa el prototipo realiza la captura del tráfico de red y se parametrizan las opciones para el funcionamiento (tiempo de captura, teléfono para notificación).

Etapa 2 - Conversión a Formato CICIDS2017: Esta etapa recibe como entrada el archivo PCAP definido en la etapa 1 y genera como salida un archivo en formato CSV con la misma estructura del conjunto de datos CICIDS2017.

Etapa 3 – Técnicas de Aprendizaje Supervisado: En la última etapa el sistema realiza la clasificación y genera los reportes sobre la predicción.

En la figura 5-8 se define la arquitectura del sistema.

Figura 5-8: Arquitectura del Prototipo



Fuente: Elaboración propia

Tal como se observa, la arquitectura del prototipo está diseñado modularmente, lo que permite utilizar alguno de los servicios de forma independiente

A continuación, se describen a detalle cada uno de los servicios expuestos en el prototipo.

5.3.1 Servicios

A. Captura de Datos:

Usando la herramienta TCPDump, se almacena el tráfico de red en un archivo .PCAP. A continuación, se observa el comando necesario para iniciar una captura parametrizada.

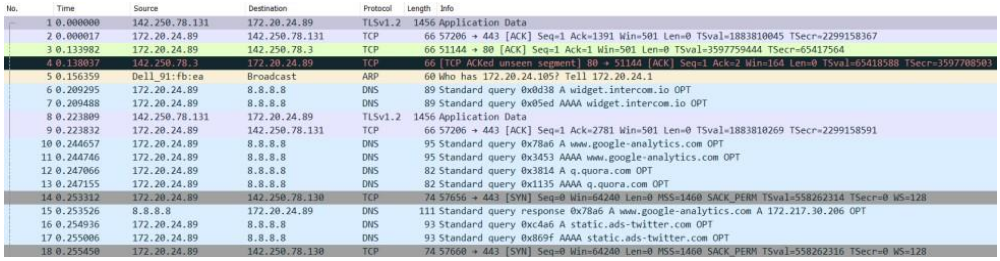
*timeout **x** tcpdump -i **y** -w **z**.pcap*

Donde:

- **x:** Tiempo de captura (1 minuto - 60 minutos máximo).

- **y:** Nombre de la interfaz de red (generado automáticamente).
- **z:** Nombre del archivo en formato .PCAP.

Figura 5-9: Estructura de un archivo .PCAP.



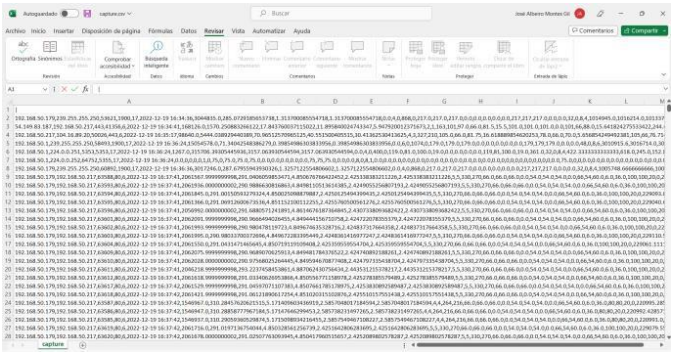
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	142.250.78.131	172.20.24.89	TLSv1.2	1456	Application Data
2	0.000017	172.20.24.89	142.250.78.131	TCP	66	57206 → 443 [ACK] Seq=1 Ack=1391 Win=501 Len=0 TSval=1883810045 TSecr=2299158367
3	0.133982	172.20.24.89	142.250.78.3	TCP	66	51144 → 80 [ACK] Seq=1 Ack=1 Win=501 Len=0 TSval=3597759444 TSecr=65417564
4	0.138037	142.250.78.3	172.20.24.89	TCP	66	[TCP ACKed unseen segment] 80 → 51144 [ACK] Seq=1 Ack=2 Win=164 Len=0 TSval=65418588 TSecr=3597708503
5	0.156359	Dell_91:fb:ea	Broadcast	ARP	60	Who has 172.20.24.105? Tell 172.20.24.1
6	0.209295	172.20.24.89	8.8.8.8	DNS	89	Standard query 0x0d38 A widget.intercom.io OPT
7	0.209488	172.20.24.89	8.8.8.8	DNS	89	Standard query 0x05ed AAAA widget.intercom.io OPT
8	0.223809	142.250.78.131	172.20.24.89	TLSv1.2	1456	Application Data
9	0.223832	172.20.24.89	142.250.78.131	TCP	66	57206 → 443 [ACK] Seq=1 Ack=2781 Win=501 Len=0 TSval=1883810269 TSecr=2299158591
10	0.244657	172.20.24.89	8.8.8.8	DNS	95	Standard query 0x78a6 A www.google-analytics.com OPT
11	0.244746	172.20.24.89	8.8.8.8	DNS	95	Standard query 0x3453 AAAA www.google-analytics.com OPT
12	0.247066	172.20.24.89	8.8.8.8	DNS	82	Standard query 0x3814 A q.quora.com OPT
13	0.247155	172.20.24.89	8.8.8.8	DNS	82	Standard query 0x1135 AAAA q.quora.com OPT
14	0.253312	172.20.24.89	142.250.78.130	TCP	74	57656 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=558262314 TSecr=0 WS=128
15	0.253526	8.8.8.8	172.20.24.89	DNS	111	Standard query response 0x78a6 A www.google-analytics.com A 172.217.30.206 OPT
16	0.254936	172.20.24.89	8.8.8.8	DNS	93	Standard query 0xc4a6 A static.ads-twitter.com OPT
17	0.255006	172.20.24.89	8.8.8.8	DNS	93	Standard query 0x869f AAAA static.ads-twitter.com OPT
18	0.255450	172.20.24.89	142.250.78.130	TCP	74	57660 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=558262316 TSecr=0 WS=128

Fuente: Elaboración propia

B. Conversión de Datos:

La conversión de datos se realizó por medio de la herramienta CICflowmeter. Este generador de tráfico permite convertir los registros a formato CSV desde un archivo. PCAP (Preuveneers et al., 2018).

Figura 5-10: Estructura de un archivo CSV convertido por CICFlowMeter.



Fuente: Elaboración propia.

A continuación, en las tablas 5-4, 5-5 y 5-6 se define la estructura del conjunto de datos CICIDS2017.

Tabla 5-4: Columnas CICIDS2017 - parte 1

Número	Columna CICIDS2017
1	dst_port
2	flow_duration
3	tot_fwd_pkts
4	tot_bwd_pkts
5	totlen_fwd_pkts
6	totlen_bwd_pkts
7	fwd_pkt_len_max
8	fwd_pkt_len_min
9	fwd_pkt_len_mean
10	fwd_pkt_len_std
11	bwd_pkt_len_max
12	bwd_pkt_len_min
13	bwd_pkt_len_mean
14	bwd_pkt_len_std
15	flow_byts_s
16	flow_pkts_s
17	flow_iat_mean
18	flow_iat_std
19	flow_iat_max
20	flow_iat_min
21	fwd_iat_tot
22	fwd_iat_mean
23	fwd_iat_std
24	fwd_iat_max
25	fwd_iat_min
26	bwd_iat_tot

Fuente: Elaboración propia

Tabla 5-5: Columnas CICIDS2017 - parte 2

Número	Columna CICIDS2017
27	bwd_iat_mean
28	bwd_iat_std
29	bwd_iat_max
30	bwd_iat_min
31	fwd_psh_flags
32	bwd_psh_flags
33	fwd_urg_flags
34	bwd_urg_flags
35	fwd_header_len
36	bwd_header_len
37	fwd_pkts_s
38	bwd_pkts_s
39	pkt_len_min
40	pkt_len_max
41	pkt_len_mean
42	pkt_len_std
43	pkt_len_var
44	fin_flag_cnt
45	syn_flag_cnt
46	rst_flag_cnt
47	psh_flag_cnt
48	ack_flag_cnt
49	urg_flag_cnt
50	cwe_flag_count
51	ece_flag_cnt
52	down_up_ratio

Fuente: Elaboración propia

Tabla 5-6: Columnas CICIDS2017 - parte 3

Número	Columna CICIDS2017
53	pkt_size_avg
54	fwd_seg_size_avg
55	bwd_seg_size_avg
56	fwd_byts_b_avg
57	fwd_pkts_b_avg
58	fwd_blk_rate_avg
59	bwd_byts_b_avg
60	bwd_pkts_b_avg
61	bwd_blk_rate_avg
62	subflow_fwd_pkts
63	subflow_fwd_byts
64	subflow_bwd_pkts
65	subflow_bwd_byts
66	init_fwd_win_byts
67	init_bwd_win_byts
68	fwd_act_data_pkts
69	fwd_seg_size_min
70	active_mean
71	active_std
72	active_max
73	active_min
74	idle_mean
75	idle_std
76	idle_max
77	idle_min
78	Label

Fuente: Elaboración propia

C. Clasificación por Medio de los Modelos:

La detección de tráfico asociado a un ataque DoS/DDoS se realiza por medio de los modelos de clasificación previamente entrenados o por las instancias de entrenamiento. A continuación, se definen las técnicas para la detección de ataques.

- ***K Vecinos Cercanos.***
- ***Red Neuronal Profunda.***
- ***Red Neuronal Convolucional.***

Inicialmente consideraron las técnicas de clasificación Máquina de Soporte Vectorial (SVM), Regresión Logística (RL) y Clasificador Bayesiano Gaussiano (Naive Bayes), no obstante, no fueron tenidas en cuenta dado su rendimiento en términos de métricas de clasificación y tiempos de respuesta al momento de reportar una clasificación.

D. Reportes:

El prototipo expone en la clasificación un reporte por medio de gráficas y un archivo en formato .PDF, la predicción definida por el modelo.

5.4 Implementación del Prototipo Basado en Aprendizaje Supervisado en Arquitectura Orientada a Servicios

La implementación del prototipo fue realizada bajo un enfoque de ciclo de vida de software tradicional, definiendo los requerimientos y el diseño desde etapas tempranas, para posteriormente realizar la codificación y la etapa de pruebas.

A continuación, se detalla cada etapa del ciclo de vida de software aplicado al desarrollo del Prototipo Basado en Aprendizaje Supervisado en Arquitectura Orientada a Servicios en la Nube.

5.4.1 Enfoques del Prototipo, Análisis de Requerimientos y Diseño.

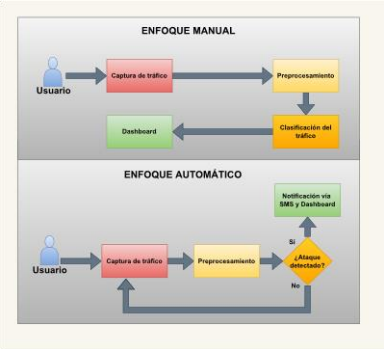
Enfoques del Prototipo

En la etapa de análisis de requerimientos y diseño se ha sido definido que el prototipo sería implementado bajo 2 enfoques; manual y automático. Esta propuesta surge de la necesidad de realizar la detección de ataques de tipo DoS/DDoS por medio de un análisis en tiempo real, sin la necesidad de parametrización del prototipo, adicionando la opción de notificación vía mensaje de texto en caso de la detección de un ataque. Por otro lado, el enfoque manual permite la captura de tráfico, conversión de tramas a formato CICIDS2017 y detección de intrusiones de forma independiente.

Enfoque Manual: Garantiza que los usuarios pueden consumir los diferentes servicios de forma separada, lo cual permite que el sistema sea usado con un mismo conjunto de datos en diferentes momentos para realizar comparaciones y análisis sobre las predicciones.

Enfoque Automático: Determina la detección de intrusiones automáticamente sin necesidad de parametrizar entradas en el prototipo, a diferencia del enfoque manual. El sistema analiza el tráfico de red en búsqueda de firmas asociadas a los ataques DoS/DDoS, y en el momento en el cual los modelos detectan un ataque, se notifica vía SMS al usuario y se generan los reportes sobre las tramas analizadas. En la figura 5-11 se pueden observar los procesos que se ejecutan en cada uno de los enfoques, lo cual ayuda al usuario a seleccionar el que más se acomode a sus necesidades en un momento determinado.

Figura 5-11: Enfoques del Prototipo



Fuente: Elaboración propia

Tal como se pudo observar en la figura 5-11, el prototipo ejecuta los mismos procesos, no obstante, algunos son realizados automáticamente hasta la detección de un ataque.

5.4.2 Listado de Requerimientos

A continuación, en la tabla 5-7 se definen los requerimientos funcionales y en la tabla 5-8 requerimientos no funcionales.

Tabla 5-7: Requerimientos Funcionales del Prototipo

#	Requerimiento	Descripción	Prioridad
RF1	Parametrizar tiempo de captura de tráfico de red	El sistema debe permitir al usuario definir un tiempo en minutos para capturar el tráfico de red (mínimo 1 – máximo 60)	5
RF2	Parametrizar interfaz de tarjeta de red	El sistema debe permitir al usuario seleccionar la tarjeta de red por la cual desea capturar el tráfico.	5
RF3	Capturar y almacenar del tráfico de red	El sistema debe permitir capturar y almacenar el tráfico de red en formato PCAP	5
RF4	Analizar, convertir, clasificar y notificar automáticamente	El sistema debe permitir al usuario almacenar el tráfico de red, seleccionar la interfaz de la tarjeta de red, seleccionar el algoritmo de clasificación, realizar la conversión a formato CICIDS2017, definir la predicción y en caso de detectar un ataque, notificar vía mensaje de texto al usuario.	5
RF5	Parametrizar el número de celular al cual será enviado el mensaje con notificación y el token API.	El sistema debe permitir recibir el número de celular al cual será enviado el mensaje de texto al momento de obtener la detección de un ataque y el token de seguridad que requiere la API para su funcionamiento.	4
RF6	Carga, conversión y almacenamiento de captura de tráfico de red en formato CSV	El sistema debe permitir la carga del tráfico de red en formato PCAP, convertirlo a la estructura del dataset CICIDS2017 y almacenarlo en formato CSV.	5
RF7	Generación de reportes	El sistema debe permitir la renderización de gráficas y reporte en formato PDF con la clasificación generada por el modelo.	5

Fuente: Elaboración propia

Tabla 5-8: Requerimientos no Funcionales del Prototipo

#	Requerimiento	Descripción	Prioridad
RNF1	Comportamiento responsive	El prototipo debe estar en la capacidad de adaptarse a diferentes dispositivos.	3
RNF2	Interfaz sencilla e intuitiva	El prototipo debe tener una interfaz clara, sin sobre carga de elementos y buen contraste.	4
RNF3	Modularización de los servicios ofrecidos	Los servicios definidos en el prototipo deben estar diseñados modularmente, lo que permite que funcionen de forma independiente.	5
RNF4	Menú de navegación	El prototipo debe tener un menú de navegación para acceder fácilmente a las diferentes opciones.	4
RNF5	Despliegue sobre ambientes Linux	El prototipo debe estar implementado para sistemas operativos Linux.	4
RNF6	Validaciones de formularios tanto del lado del cliente como del lado del servidor	Las validaciones de los formularios deben hacerse desde el lado del cliente por medio de HTML5 y JavaScript, al igual que del lado del servidor, para aumentar los niveles de seguridad del prototipo.	5
RFN7	Documentación del código	El sistema debe contar con documentación del código para garantizar facilidad en el mantenimiento y mejoras.	3
RNF8	Incluir prácticas de accesibilidad web y glosario	El prototipo debe tener formularios con características mínimas de accesibilidad (nivel A), al igual que un glosario para asociar términos y definiciones.	4

Fuente: Elaboración propia

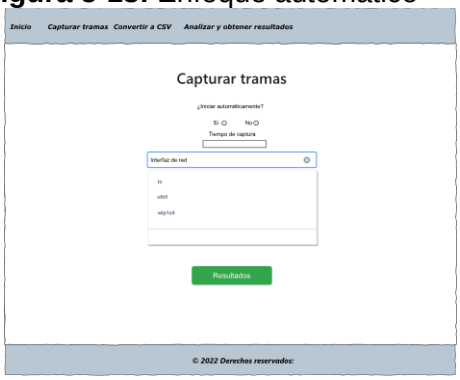
5.4.3 Diseño

Teniendo en cuenta las funcionalidades definidas en las tablas 5-8 y 5-7, se definieron algunos bosquejos para facilitar el *frontend* del prototipo (ver figuras 5-12 hasta 5-12).

Figura 5-12: Vista de inicio (home)

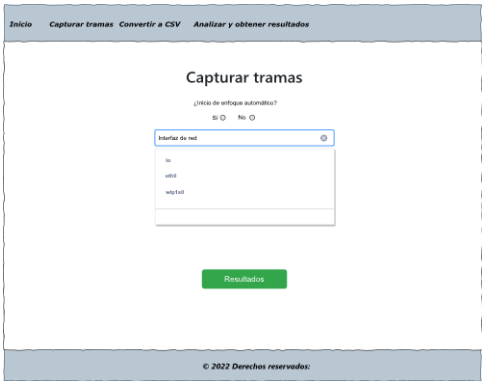
Fuente: Elaboración propia

Figura 5-13: Enfoque automático



Fuente: Elaboración propia

Figura 5-14: Inicio de captura modo manual



Fuente: Elaboración propia

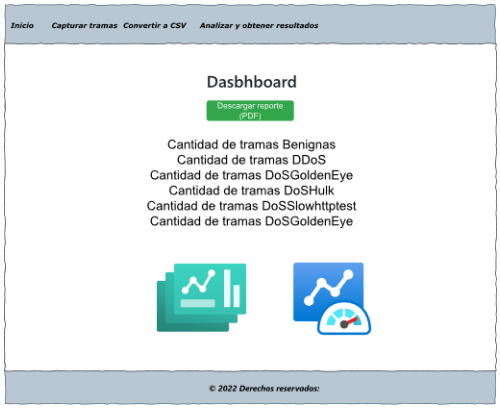
Figura 5-15: Cargar archivo PCAP

Fuente: Elaboración propia

Figura 5-16: Selección de técnicas de Aprendizaje Supervisado

Fuente: Elaboración propia

Figura 5-17: Visualización de dashboard



Fuente: Elaboración propia

5.5 Implementación

La etapa de implementación fue ejecutada en el ambiente definido en la tabla 5-9.

Tabla 5-9: Resumen de Tecnologías

Finalidad	Tecnología
Backend para servicios web	Flask
Frontend para servicios web	HTML, CSS, JavaScript, Bootstrap
Creación de modelos para clasificación	Colab
Captura de tráfico de red	TCPDump
Conversión de tráfico a formato CSV	CICFlowmeter
Virtualización de Python	Pyenv: 3.8.5 y 3.7.13
Reportes (Gráficos y PDF)	Plotly JS y PDFKit
Notificación vía MSM	Twilio API
Sistema Operativo (host)	Kali Linux

Fuente: Elaboración propia

Flask: Micro framework basado en Python que permite realizar aplicaciones web bajo diferentes arquitecturas sin la necesidad de ocuparse de detalles de bajo nivel como la gestión de protocolos (Singh et al., 2022).

CICFlowMeter: Herramienta implementada por el Instituto Canadiense de Ciberseguridad para la creación de conjuntos de datos en formato CSV a partir de tráfico capturado en formato PCAP. De allí se generan en total 79 columnas (una de ellas es eliminada dado que está repetida), las cuales corresponden con las mismas definidas en el dataset CICIDS2017 (Preuveneers et al., 2018).

Reportes: Para la generación de las gráficas se usó la librería Plotly.js, escrita en JavaScript (Gholizadeh, 2022). Los datos son recibidos en formato JSON (Notación de Objeto de JavaScript) y se renderizan en un elemento en un documento HTML. El reporte en formato PDF fue diseñado por medio PDFKit, una librería implementada en JavaScript, la cual convierte documentos HTML a PDF.

Pyenv: Teniendo en cuenta que el algoritmo KNN usa la librería de aceleración FAISS, la cual requiere Python 3.7.13, es necesario definir una estrategia para la virtualización de 2 ambientes virtuales (3.8.5 y 3.7.13). Pyenv permite tener múltiples versiones de Python con diferentes versiones para las dependencias necesarias en el prototipo.

Kali Linux: El sistema operativo Kali Linux corresponde a una distribución basada en Debian el cual tiene un enfoque orientado a la seguridad informática, dada su integración con herramientas para el análisis de tráfico, informática forense, hacking ético, entre otras (Cisar & Pinter, 2019).

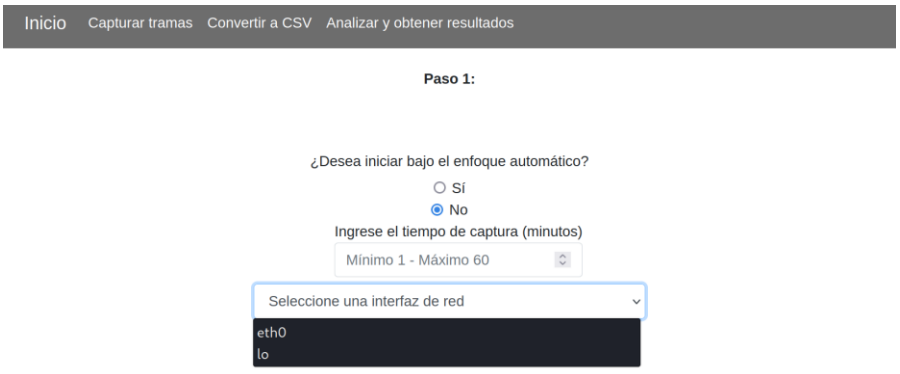
Twilio API: El envío de notificaciones vía sms se realiza por medio de la Interfaz de Programación de Aplicaciones (API) de Twilio. Se definen diferentes servicios de comunicación, y entre ellos se encuentra el envío de mensajes de texto a teléfonos celulares (S et al., 2019). El precio por cada mensaje es de $0.0525. A los nuevos usuarios se les otorga una versión de prueba correspondiente a $15 dólares.

TCPDump: Herramienta que permite guardar en formato PCAP el tráfico de red capturado. El uso de esta tecnología se determinó teniendo en cuenta la recomendación de algunos autores, los cuales concluyen que TCPDump puede ser una mejor alternativa cuando el ancho de banda supera las 200 Mbit/s (Lachnit et al., 2021).

Funcionalidades del Prototipo - Enfoque Manual

Parametrización de captura: Tal como se observa en la figura 5-18, al seleccionar el enfoque manual, es necesario parametrizar el prototipo en términos de interfaz de red y tiempos de captura.

Figura 5-18: Selección de Enfoque Manual



Fuente: Elaboración propia

Dado que el prototipo tiene en cuenta las interfaces de red de forma dinámica, siempre se mostrarán todas las interfaces habilitadas en el equipo donde se despliegue el sistema.

Selección de Captura de Tráfico (archivo PCAP): Luego de terminada la captura del tráfico de red, el prototipo solicita al usuario la carga del archivo en el cual se almacenaron las tramas en formato PCAP, tal cual como se observa en la figura 5-19.

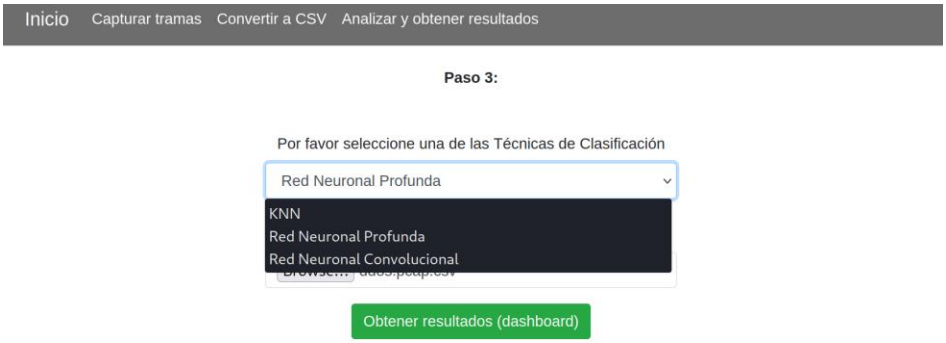
Figura 5-19: Carga de captura de red (Archivo PCAP)

Fuente: Elaboración propia

La funcionalidad observada en la figura 5-19 genera un archivo en formato CSV convertido al formato del dataset CICIDS2017.

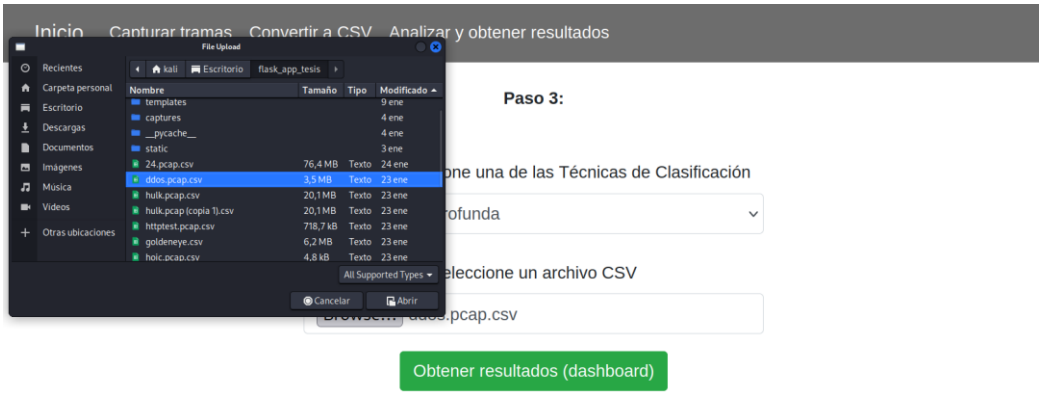
Selección del Archivo CSV (Formato dataset CICIDS2017) y Selección de Técnica de Aprendizaje Supervisado: En las figuras 5-21 y 5-20 se observa que es necesario seleccionar la técnica de clasificación para obtener los resultados sobre la predicción, así como la carga del archivo en formato CSV generado en la funcionalidad anterior.

Figura 5-20: Selección de Técnica de Aprendizaje Supervisado



Fuente: Elaboración propia

Figura 5-21: Carga del archivo CSV (en el formato del dataset CICIDS2017)



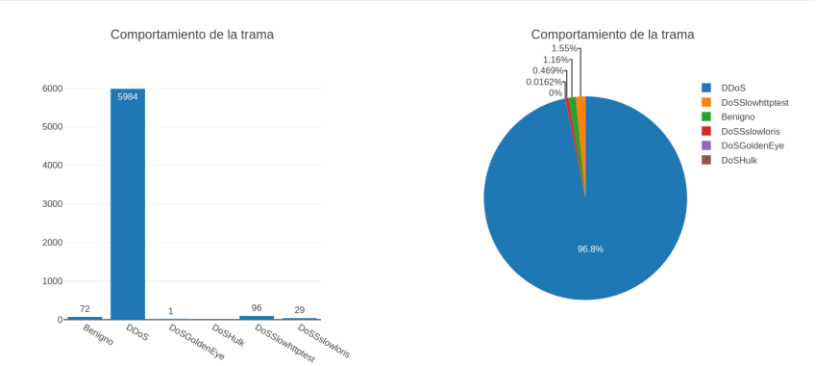
Fuente: Elaboración propia

Generación de Reportes (dashboard y PDF): Luego de la carga del archivo en formato CSV y de la selección de la técnica de clasificación, el prototipo genera un *dashboard* con gráficas sobre el comportamiento de la trama y un archivo en formato PDF con un resumen sobre la predicción (ver figuras 5-22, 5-23, 5-24 y 5-25).

Figura 5-22: Reporte Enfoque Manual (parte 1)

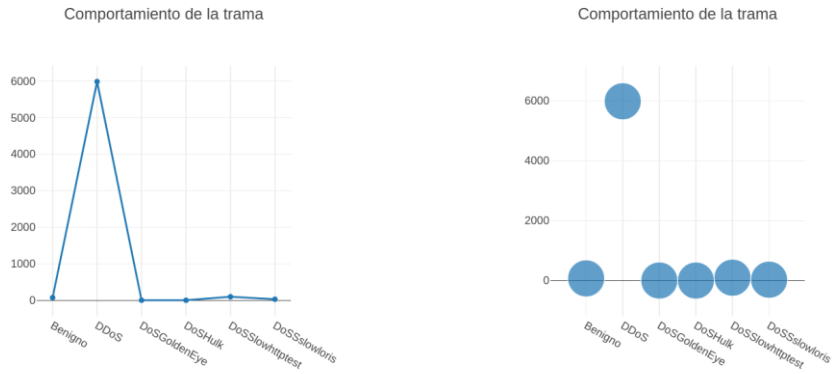
Fuente: Elaboración propia

Figura 5-23: Reporte Enfoque Manual (parte 2)



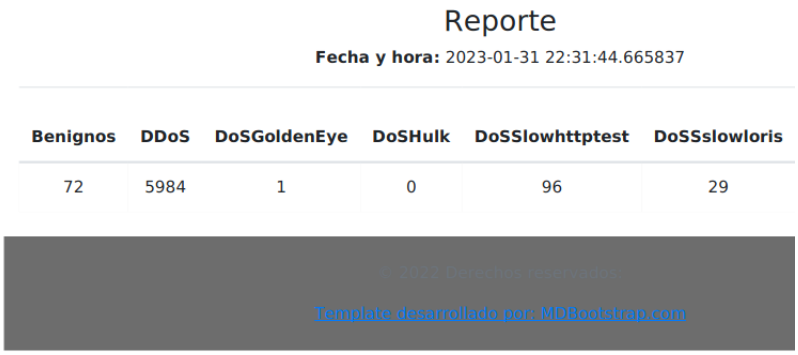
Fuente: Elaboración propia

Figura 5-24: Reporte Enfoque Manual (parte 3)



Fuente: Elaboración propia

Figura 5-25: Reporte Enfoque Manual (parte 4)

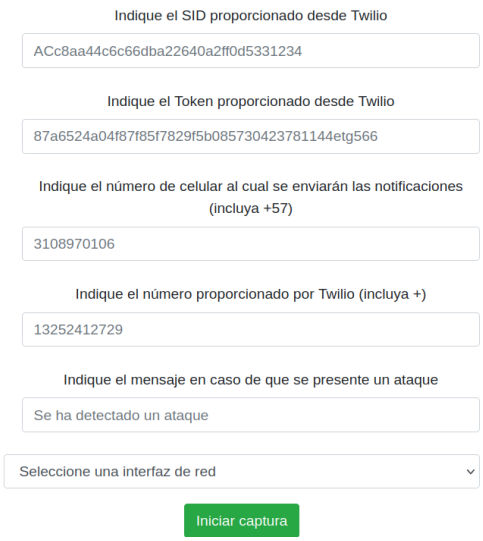


Fuente: Elaboración propia

Funcionalidades del Prototipo Enfoque Automático

Inicio de Monitoreo Automático: Este enfoque requiere la parametrización por medio de Twilio (ver figura 5-26), un servicio de mensajería de texto, el cual permite enviar una notificación al usuario en caso de que se presente tráfico clasificado como ataque. La personalización de este enfoque se explica a detalle en la sesión de despliegue, en el presente capítulo.

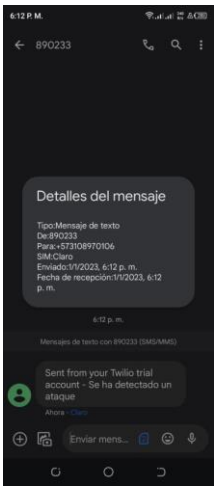
Figura 5-26: Parametrización del Prototipo (enfoque automático)



Fuente: Elaboración propia

Después de la configuración del prototipo, se inicia una captura de tráfico continúa, y en el momento en el cual se detecte un ataque, se enviará un mensaje de texto con el mensaje determinado por el usuario en la parametrización (ver figura 5-27).

Figura 5-27: Notificación vía SMS



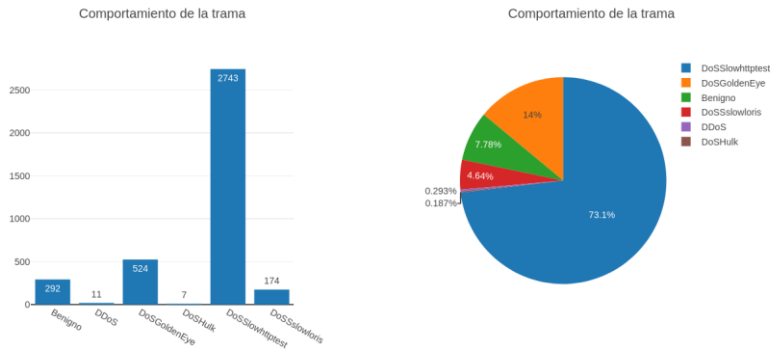
Fuente: Elaboración propia

En la figura 5-28, se observa que posterior al envío mensaje, se genera el reporte con el análisis del tráfico de red, tal como ocurre en el enfoque manual.

Figura 5-28: Reporte Enfoque Automático (parte 1)

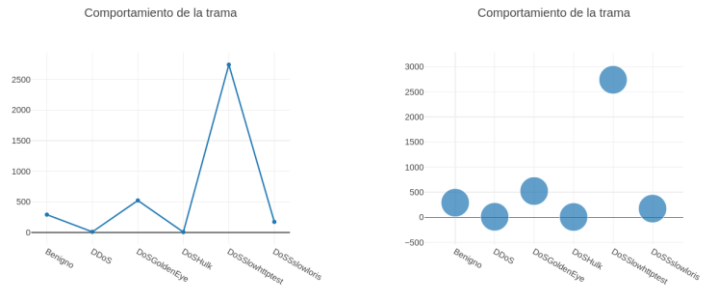
Fuente: Elaboración propia

Figura 5-29: Reporte Enfoque Automático (parte 2)



Fuente: Elaboración propia

Figura 5-30: Reporte Enfoque Automático (parte 3):



Fuente: Elaboración propia

Figura 5-31: Reporte Enfoque Automático (parte 4):

Reporte

Fecha y hora: 2023-01-31 22:40:40.602378

Benignos	DDoS	DoSGoldenEye	DoSHulk	DoSSlowhttptest	DoSSslowloris
292	11	524	7	2743	174

Fuente: Elaboración propia

5.6 Pruebas

Para verificar el correcto funcionamiento del prototipo se definió el ambiente de prueba mostrado en la tabla 5-10.

Tabla 5-10: Ambiente de prueba del prototipo

Lugar	Laboratorio LIGRED (Universidad Nacional)
Cantidad de usuarios	2
Cantidad de equipos	3 (2 Windows – 1 Ubuntu)
Dispositivo de conexión	Switch capa 2
Cableado	Categoría 5
Duración de las pruebas	(45 minutos – 1 hora)
Fecha	21/12/2023
Descripción	Evaluación para observar el funcionamiento del prototipo por medio del análisis en red local.
Conclusiones	El prototipo demostró un correcto funcionamiento durante la prueba, sin verse comprometido en ningún momento.

Fuente: Elaboración propia

Figura 5-32: Ambiente de pruebas para verificar funcionamiento del prototipo - parte 1

Fuente: Elaboración propia

Durante la prueba para verificación del funcionamiento del prototipo, 2 estudiantes del programa curricular Administración de Sistemas Informáticos de la Universidad Nacional de Colombia realizaron visitas a diferentes sitios web, como Stack Overflow, portales de noticias, sitios web institucionales de la universidad y GitHub.

Figura 5-33: Ambiente de pruebas para verificar funcionamiento del prototipo - parte 2

Fuente: Elaboración propia

En la segunda parte del experimento, las estudiantes visitaron los mismos sitios web de la prueba 1, adicionando peticiones HTTP desde un aplicativo web implementado en Laravel, el cual fue desplegado localmente en una de las máquinas de la red.

El objetivo de la prueba no fue analizar el rendimiento en términos de métricas de los modelos, sino, de confirmar el correcto funcionamiento del prototipo en función de tiempos de captura.

5.7 Despliegue

Para el correcto funcionamiento del prototipo, a continuación, se definen los pasos del despliegue del prototipo.

Sistema Operativo: Para el correcto funcionamiento del prototipo es necesario ejecutar la instalar sobre un sistema operativo con una distribución Linux. Adicionalmente, es importante realizar la instalación de 2 ambientes de Python por medio de Pyenv (3.7.13 y 3.8.5), al igual que TCPDump y Git.

Repositorio GitHub: El código fuente se encuentra expuesto en un repositorio de GitHub. Enlace: https://github.com/joamontesgi/flask_app_tesis

Se debe clonar el repositorio usando el comando git clone.

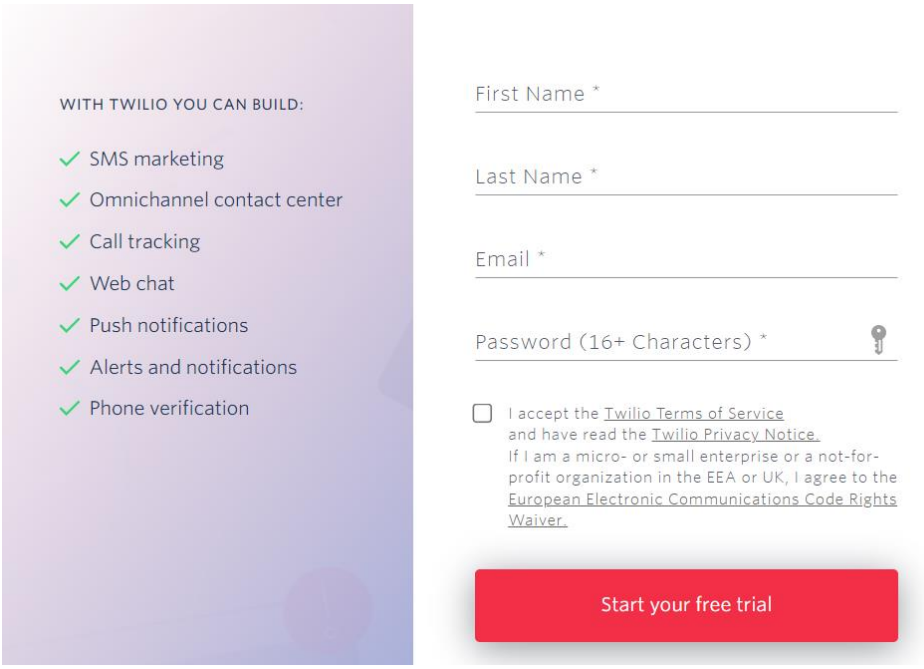
Instalación de Dependencias: Para instalar las dependencias necesarias es necesario ejecutar el comando pip install -r requirements.txt.

Navegador web: Es necesario instalar un navegador web para visualizar la renderización de los servicios.

Configuración de Twilio: Para obtener el envío de mensaje de texto al teléfono celular es necesario crear una cuenta en Twilio, tal como se observa en la figura 5-34.

Enlace: https://www.twilio.com/try-twilio

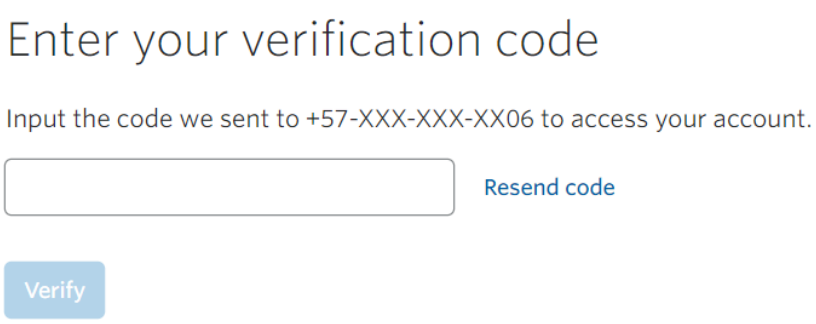
Figura 5-34: Creación de cuenta en Twilio



Fuente: Elaboración propia

Luego de la verificación del correo electrónico, es posible iniciar la versión de prueba, tal como se observa en la figura 5-35. Se debe indicar el número de teléfono celular en el cual se desean recibir los mensajes de texto con la notificación en caso de la detección de ataques.

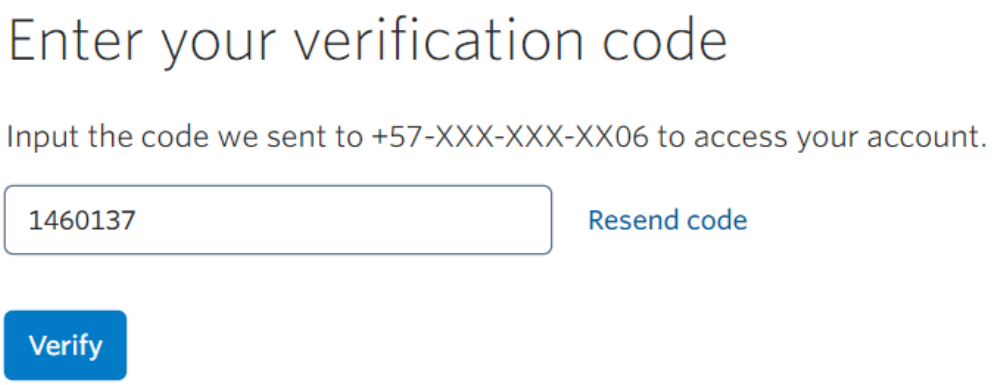
Figura 5-35: Número de teléfono celular para recibir notificaciones



Fuente: Elaboración propia

Para continuar con el proceso, es necesario ingresar el código de verificación enviado al celular asignado, tal como se observa en la figura 5-36.

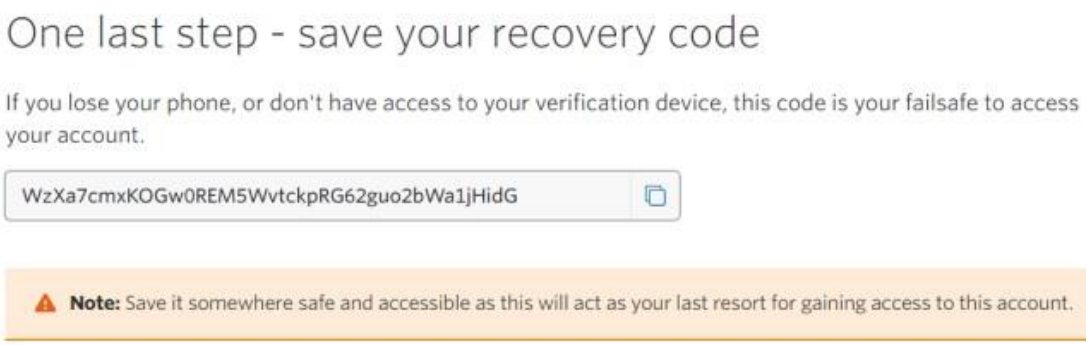
Figura 5-36: Código de verificación Twilio



Fuente: Elaboración propia

Adicionalmente, se recomienda guardar el token de verificación proporcionado (ver figura 5-37).

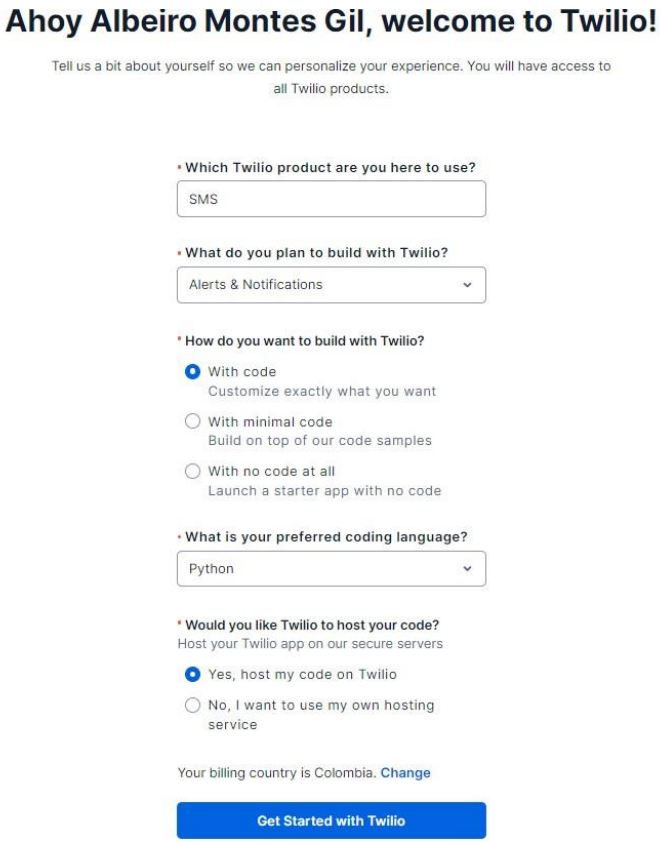
Figura 5-37: Token de verificación



Fuente: Elaboración propia

Se recomienda terminar el registro con las opciones indicadas en la figura 5-38.

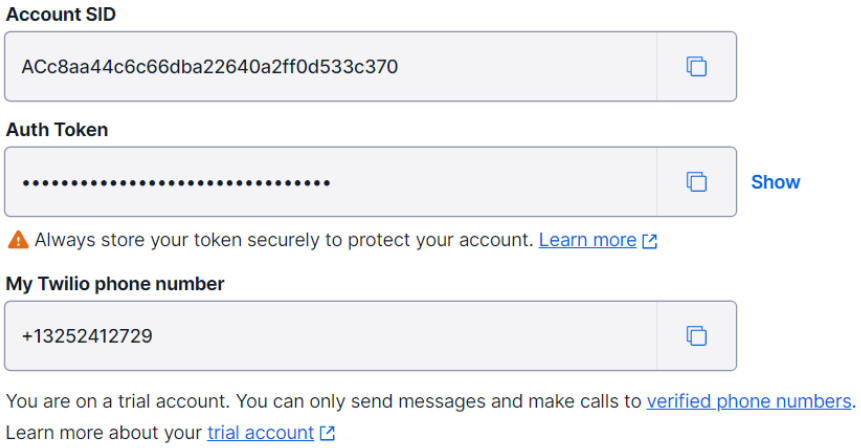
Figura 5-38: Formulario de Registro



Fuente: Elaboración propia

Por último, Twilio permite mostrar el token de autenticación, la cuenta SID y el número teléfono desde el cual serán enviados los mensajes (ver figura 5-39).

Figura 5-39: Información de la cuenta



Fuente: Elaboración propia

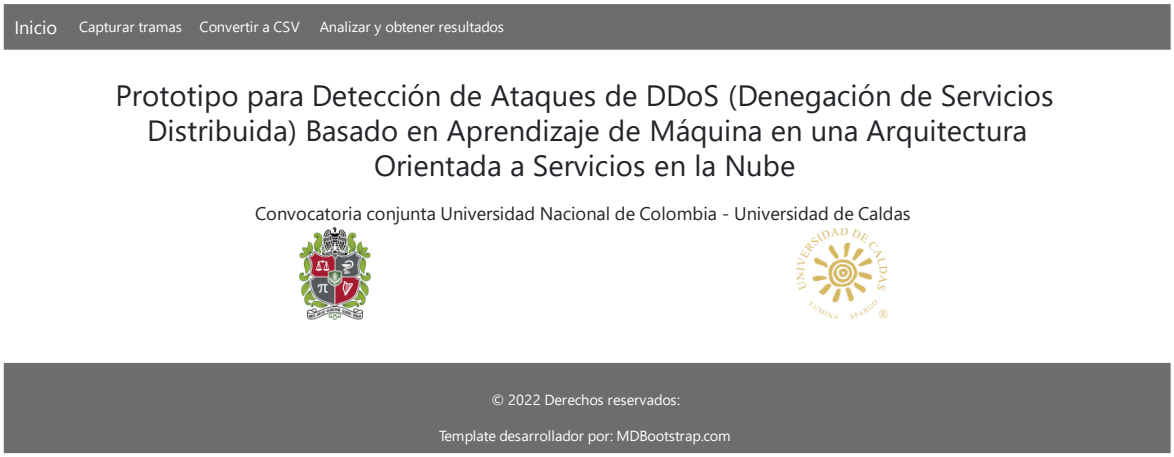
Inicio del Servidor: Desde una terminal ubicada en el directorio raíz del proyecto, se debe ejecutar el comando python app.py tal como se observa en la figura 5-40 y 5-41. Esto iniciará el servidor proporcionado por Flask para la ejecución del prototipo en el navegador, por medio de la URL http://127.0.0.1:5000.

Figura 5-40: Iniciando servidor

```
@joamontesgi →/workspaces/flask_app_tesis (main X) $ python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deploym
ent. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 107-131-046
```

Fuente: Elaboración propia

Figura 5-41: Vista de inicio



Fuente: Elaboración propia

5.8 Conclusiones del Capítulo

En este capítulo se presentaron los principales aspectos de las etapas del ciclo de vida del software; análisis, diseño, implementación y pruebas aplicadas en la propuesta del prototipo orientado a servicios para la detección de ataques DDoS/DoS.

El enfoque manual del prototipo está diseñado para consumir los diferentes servicios de manera aislada, lo cual le permite al usuario realizar capturas y analizar el tráfico de red con diferentes técnicas.

El enfoque automático se diseñó de acuerdo con la necesidad de analizar el tráfico de red en tiempo real. En el momento en el cual el prototipo realice la detección de un ataque, se envía una notificación al administrador de red vía mensaje de texto para que se tomen las medidas necesarias de manera oportuna.

El ambiente de pruebas determinó que el tráfico broadcast no es lo suficientemente grande para generar predicciones acertadas, por lo que es importante realizar un monitoreo constante en el equipo donde esté desplegado el prototipo. Adicionalmente, el prototipo demostró un comportamiento adecuado en términos de funcionamiento del prototipo para las mediciones efectuadas.

El prototipo desarrollado está orientado a la detección de ataques de tipo DDoS/DoS, sin embargo, es posible adaptarlo a nuevas formas de ataques, definiendo una nueva etapa de entrenamiento con las características de otros ataques.

El principal ambiente de pruebas fue ejecutado bajo el sistema operativo Kali Linux, sin embargo, el prototipo puede ser instalado en otras distribuciones de Linux siempre que se garantice el correcto ambiente de despliegue, el cual se compone de las dependencias definidas en los archivos requirements.txt ubicados en el repositorio y las versiones de Pyenv necesarias.

6 Capítulo 6: Verificación y validación del IDS orientado a servicios en la nube para la detección de ataques de denegación de servicios distribuidos.

En este capítulo se presenta la verificación y validación realizada al Sistema de Detección de Intrusos presentado en el capítulo 5. En primera instancia se diseña cada una de las pruebas ejecutadas. En segunda instancia se encuentran los detalles técnicos de cada una de las pruebas. En última instancia se relacionan algunos gráficos para facilitar el análisis de los resultados a partir de las validaciones generadas.

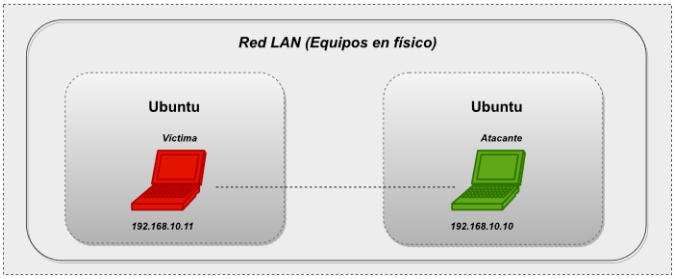
Teniendo en cuenta que los algoritmos de clasificación fueron entrenados y validados por medio del conjunto de datos CICIDS2017, tal como se observó en el capítulo 4 de esta tesis, buscando atacar una de las debilidades más frecuentes en los trabajos relacionados y expuestos en el estado del arte, se han realizado pruebas de validación y verificación con tráfico real y usuarios.

Los ambientes de pruebas definidos estuvieron enfocados al análisis y comportamiento del prototipo bajo ambientes controlados, en los cuales se realizaron lanzamientos de ataques en subredes virtuales y físicas. A continuación, se definen todos los diseños para la validación del prototipo.

6.1 Diseño del ambiente de pruebas para validación del Sistema.

6.1.1 Prueba 1: Validación con subred de 2 equipos.

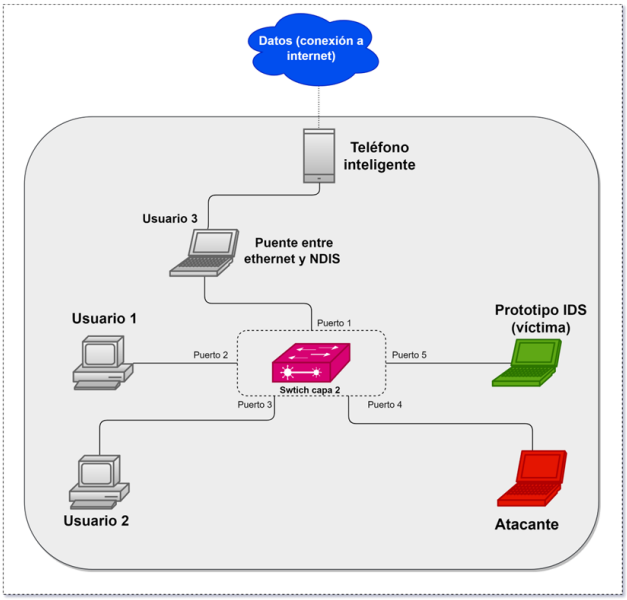
Figura 6-1: Prueba 1 (2 equipos físicos en subred)



Fuente: Elaboración propia

6.1.2 Prueba 2: Validación con subred de equipos en físico y usuarios.

Figura 6-2: Prueba 2 (3 usuarios y equipos reales)



Fuente: Elaboración propia

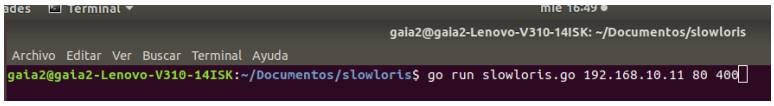
6.2 **Ataques**

A continuación, se explican a detalle los ataques usados en la validación del prototipo de IDS. Se realiza una breve descripción del comportamiento general de cada ataque, se expone la manera en la cual fueron ejecutados y un resumen sobre el comportamiento en la máquina víctima.

6.2.1 Ataque DoSSlowloris.

Este ataque se caracteriza por un envío de paquetes livianos pero constante, lo que puede generar problemas a mediano y largo plazo en la disponibilidad del servicio. Su principal característica es que realiza envío de solicitudes durante un tiempo corto, define un tiempo para nuevamente generar un nuevo envío de peticiones. A continuación, en la figura 6-4 se observan los parámetros definidos para el lanzamiento del ataque. En la figura 6-5 se observa el rendimiento del sistema en términos de tráfico de red por medio del visor de recursos del sistema operativo víctima.

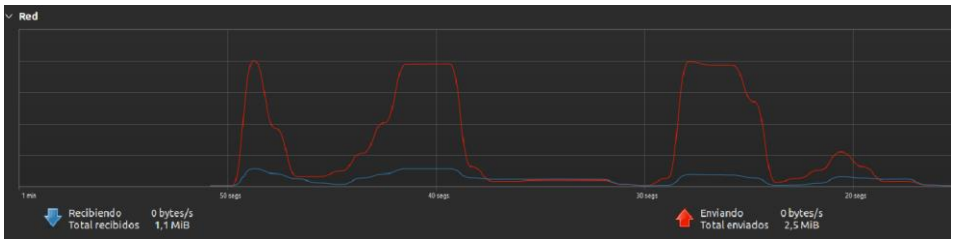
Figura 6-3: Parámetros de lanzamiento de ataque *Slowloris*.



Fuente: Elaboración propia

Tal como se puede observar en la figura 6-5, para el lanzamiento del ataque es necesario instalar el lenguaje de programación Go, conocer la dirección IP del equipo víctima, el puerto al cual se desea lanzar el ataque, en este caso el 80, y por último, la cantidad de conexiones a realizar.

Figura 6-4: Rendimiento del sistema al momento del ataque *Slowloris*



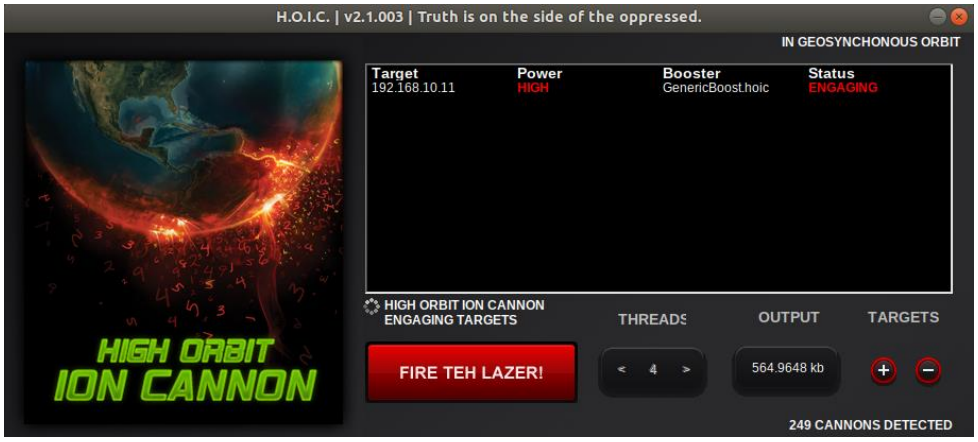
Fuente: Elaboración propia

Tal como se puede apreciar en la figura 6-5, el envío de paquetes se realiza en unos períodos de tiempo cortos y sus tramas no son muy largas, no obstante, el envío constante de estas peticiones puede generar saturación en el equipo víctima.

6.2.2 Ataque DDoS (High Orbit Ion Cannon)

El ataque generado por la herramienta HOIC otorga peticiones que pueden ser clasificadas como DDoS/DoS ataques, tal como se observa en (Black & Kim, 2022). Para el lanzamiento de un ataque por medio de la herramienta *HOIC,* es necesario conocer la dirección IP del equipo víctima, definir la potencia del ataque, la cantidad de hilos (*Threads*) y la selección de un potenciador (*Booster*), tal como se observa en la figura 6-6.

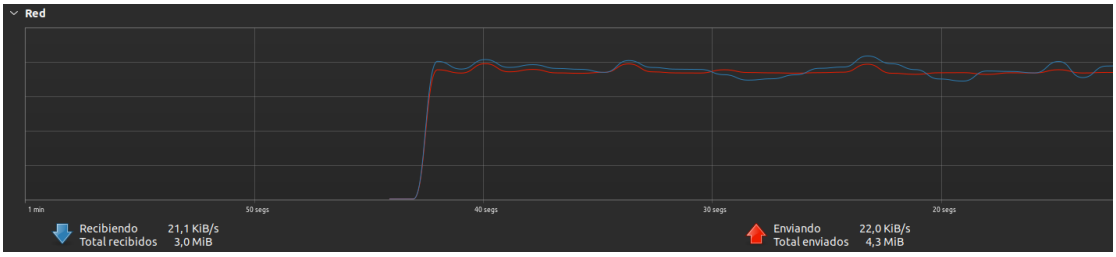
Figura 6-5: Lanzamiento de ataques por medio de *HOIC*



Fuente: Elaboración propia

En la figura 6-7 se observa el desempeño de las tramas en el equipo víctima por medio del visualizador de recursos.

Figura 6-6: Rendimiento del sistema al momento del ataque por medio de *HOIC*



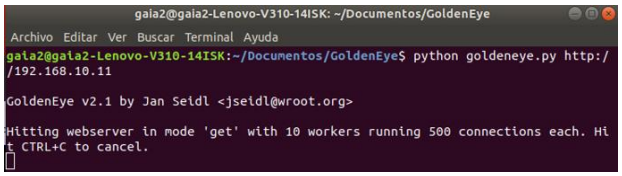
Fuente: Elaboración propia

Tal como se observa en la figura 6-6, se genera un incremento en el tráfico de red en comparación con el ataque *Slowloris*, dado que la herramienta *HOIC* genera envío de solicitudes de manera constante.

6.2.3 Ataque DoSGoldeneye

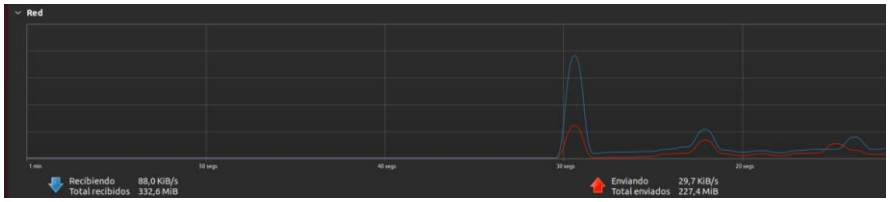
En (Beechey et al., 2023) se mencionan los ataques Goldeneye por su facilidad de lanzamiento y eficacia al momento de generar interrupciones en diferentes servicios. En la figura 6-7 se definen los parámetros para su lanzamiento. En este caso, se debe usar el protocolo http y conocer la dirección IP del equipo víctima.

Figura 6-7: Lanzamiento de ataque Goldeneye



Fuente: Elaboración propia

Figura 6-8: Rendimiento del sistema al momento del ataque Goldeneye



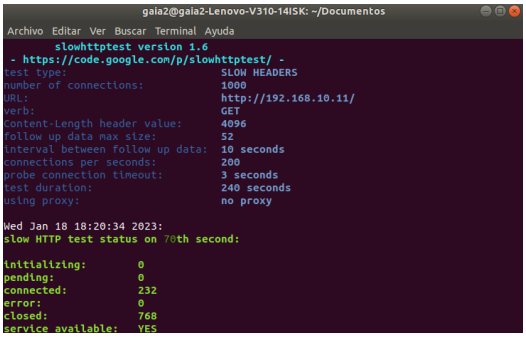
Fuente: Elaboración propia

Tal como se pudo observar en la figura 6-8, este ataque se caracteriza por un gran envío de paquetes en primera instancia, seguido de solicitudes recurrentes pero más livianas.

6.2.4 Ataque DoSSlowHTTPTest

Este ataque puede instalarse fácilmente a partir de la documentación oficial expuesta en https://www.kali.org/tools/slowhttptest/. Este tipo de ataque simula el comportamiento de las tramas DDoS/DoS en la capa de aplicación. En la figura 6-9 se define su parametrización.

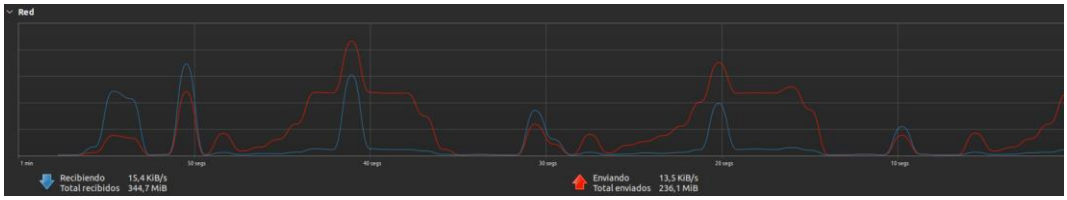
Figura 6-9: Parametrización de ataque SlowHTTPTest



Fuente: Elaboración propia

Tal como se pudo observar en la figura 6-9, este ataque requiere conocer la IP del equipo a ser atacado, el número de conexiones, tiempos de espera y longitud máxima de los paquetes. En la figura 6-10 se puede observar el comportamiento del tráfico de red al momento del lanzamiento del ataque.

Figura 6-10: Rendimiento del sistema al momento del ataque SlowHTTPTest



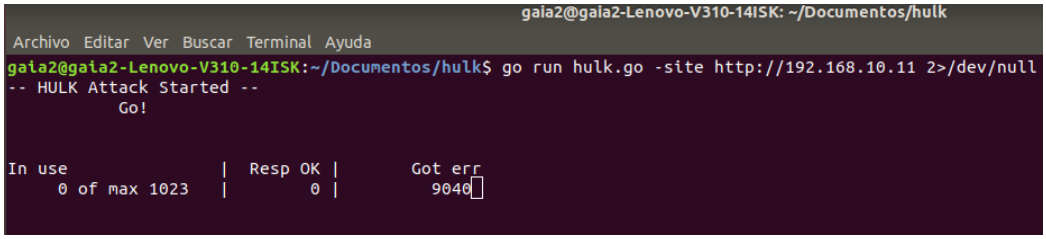
Fuente: Elaboración propia

En la figura 6-10 se observa que el comportamiento de este ataque es cambiante, lo cual se debe a los tiempos de espera que deben ser definidos en la parametrización. Se pueden encontrar picos con tramas bastante altas y otras más livianas.

6.2.5 Ataque DoSHulk

En (Benza¨ıdbenza¨ıd et al., 2020) usan el ataque Hulk para validar un sistema de protección contra ataques de tipo DDoS. Allí los autores resaltan la gran cantidad de solicitudes de tipo HTTP que genera esta técnica de ataque. A continuación, en la figura 6-11, se observa el lanzamiento de un taque Hulk cuya implementación se realizó en el lenguaje Go.

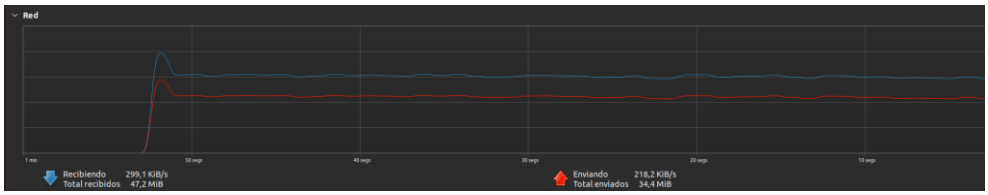
Figura 6-11: Lanzamiento de ataque Hulk



Fuente: Elaboración propia

Tal como se observa en la figura 6-12, este ataque realiza un gran envío de paquetes de manera constante, desde el

Figura 6-12: Rendimiento del sistema al momento del ataque Hulk



Fuente: Elaboración propia

Tal como se observa en la figura 6-12, este ataque realiza un gran envío de paquetes de manera constante, el cual genera una gran inundación de paquetes en el equipo receptor.

6.3 Ambientes de Validación

Se definieron 2 ambientes de validación, tal como se observa en las figuras 6-1 y 6-2, en las instalaciones de la Universidad Nacional de Colombia sede Manizales, en el Laboratorio de Investigación en Gestión de Recursos Educativos Digitales (LIGRED) y en la oficina de la Maestría en Administración de Sistemas Informáticos. Cabe resaltar que para garantizar la efectividad de los ataques, el equipo víctima no contaba con mecanismos de seguridad como antivirus ni muros cortafuegos (*firewall*).

Para este primer ambiente de validación, se estableció una prueba con entorno controlado conformado por 2 equipos portátiles, los cuales se conectaron a una red LAN por medio de un cable UTP categoría 6, tal como se observa en la figura 6-13. En esta primera prueba, el prototipo analizó las tramas de red durante un período de 5 minutos.

Figura 6-13: Ambiente para prueba 1 - Oficina de Maestría en Administración de Sistemas Informáticos

Fuente: Elaboración propia

A continuación, en la tabla 6-1 se resumen las características de cada uno de los equipos utilizados en la prueba.

Tabla 6-1: Características de los equipos usados en la prueba 1

Características	Atacante	Víctima
IP	192.168.10.10	192.168.10.11
Sistema Operativo	Ubuntu 18.04 LTS	Ubuntu 22.04 LTS
Memoria RAM	4 GB	8 GB
Marca	Lenovo V310	Lenovo Thinkbook
Procesador	Intel Core i3	AMD Ryzen 5
Tiempo de duración por ataque	5 minutos	

Fuente: Elaboración propia

Para la prueba 2 se definió un ambiente compuesto de un teléfono inteligente que compartía red de datos a uno de los 5 equipos de la red, tal como se observa en la figura 6-2 y 6-14. El principal objetivo de esta prueba es determinar si el tráfico *broadcast* propagado en la red gracias al switch no configurable de capa 2, afecta la clasificación de los modelos al momento de recibir un ataque. El comportamiento por parte de los usuarios consistió en visitar sitios web considerados como benignos, de los cuales se resaltan:

- WhatsApp.
- Instagram.

- Sistema de Información Académica de la Universidad Nacional.
- Sitios web de videojuegos en línea.
- OpenAI (ChatGPT).
- Portal web de la Universidad Nacional de Colombia.
- Sitios web de empresas de aviación.

La duración del análisis de las tramas de red por parte del prototipo fue de 3 minutos por cada ataque, dada la disponibilidad de tiempo de los usuarios. A continuación, en la tabla 6-2 se definen las características de los equipos usados en la prueba.

Tabla 6-2: Características de equipos usados en la prueba 2

Características	Equipo 1	Equipo 2	Equipo 3	Equipo víctima	Equipo atacante
IP	192.168.137.214	192.168.137.199	192.168.137.1	192.168.137.163	192.168.137.95
Sistema Operativo	Windows 10	Ubuntu 22.04 LTS	Windows 11	Ubuntu 22.04 LTS	Ubuntu 18.04 LTS
Memoria RAM (GB)	12	8	16	8	4
Marca	Dell	Lenovo	Dell	Lenovo Thinkbook	Lenovo V310
Procesador	Intel Core i7	Intel Core i7	Intel Core i7	AMD Ryzen 5	Intel Core i3
Tiempo de duración por ataque	3 minutos				

Fuente: Elaboración propia

Figura 6-14: Ambiente para prueba 2 - Laboratorio LIGRED

Fuente: Elaboración propia

6.4 Resultados Prueba 1

Desde la tabla 6-3 hasta la tabla 6-7 se definen los resultados para la prueba 1.

Tabla 6-3: Resultados ataque DoSSlowloris

Ataque DoSSlowloris							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	0	0	285	0	116	0	401
DNN	1	0	38	0	0	362	401
CNN	5	0	0	0	0	396	401

Fuente: Elaboración propia

Tabla 6-4: Resultados ataque DDoS

Ataque DDoS							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	2990	0	119	4	0	6747	9860
DNN	165	9616	0	1	65	13	9860
CNN	129	9722	9	0	0	0	9860

Fuente: Elaboración propia

Tabla 6-5: Resultados ataque DoSGoldeneye

Ataque DoSGoldeneye							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	1781	0	9821	133	2555	368	14658
DNN	58	14	11762	20	2428	376	14658
CNN	89	120	11708	15	90	2636	14658

Fuente: Elaboración propia

Tabla 6-6: Resultados ataque DoSSlowHTTPTest

Ataque DoSSlowHTTPTest							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	1789	0	1632	14	261	55	3751
DNN	141	5	494	4	2885	222	3751
CNN	292	11	524	7	2743	174	3751

Fuente: Elaboración propia

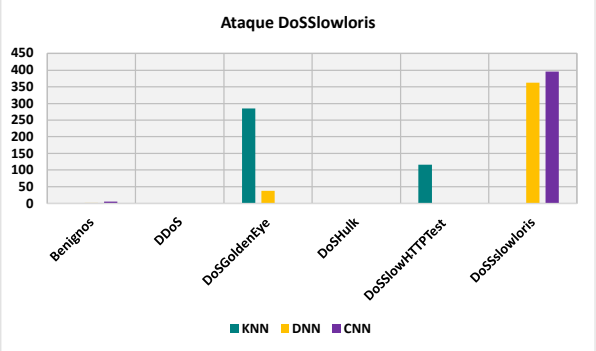
Tabla 6-7: Resultados ataque DoSHulk

Ataque DoSHulk							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	24471	49501	28241	2068	1284	2701	108266
DNN	16887	12122	567	77990	39	661	108266
CNN	24762	12995	1062	69145	168	134	108266

Fuente: Elaboración propia

A continuación, en las figuras 6-15 hasta 6-20 se observa el comportamiento de cada uno de los modelos a partir de los ataques generados.

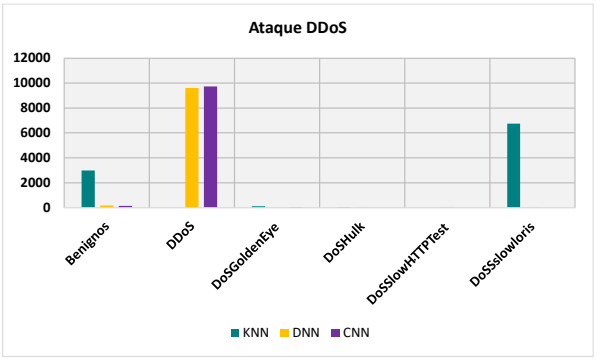
Figura 6-15: Resultados clasificación ataque DoSSlowloris



Fuente: Elaboración propia

Tal como se observa en la figura 6-15, el ataque DoSSlowloris genera una cantidad de tráfico pequeño en comparación con otros ataques, no obstante, a pesar de su tamaño, es posible identificar el tipo de tráfico por medio de los modelos CNN y DNN.

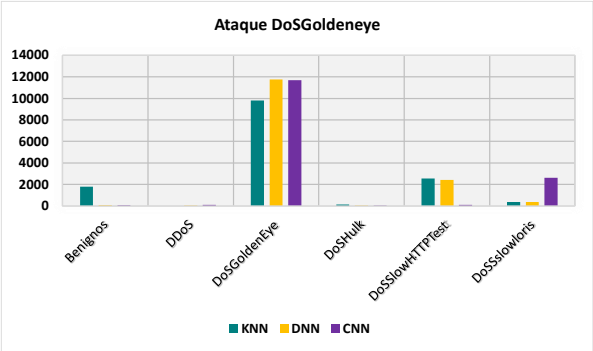
Figura 6-16: Resultados clasificación ataque DDoS



Fuente: Elaboración propia

En la figura 6-16 se observa que el número de instancias generadas por la herramienta HOIC para obtener ataques de tipo DDoS es mucho mayor en comparación al ataque DoSSlowloris, lo cual puede generar afectaciones a los equipos víctimas dada la gran cantidad de solicitudes en un corto período de tiempo. Las predicciones obtenidas por los modelos CNN y DNN fueron bastantes consistentes con respecto al tipo de ataque generado, a diferencia del clasificador KNN.

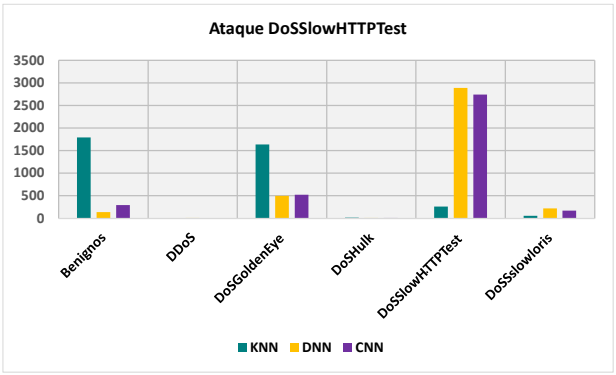
Figura 6-17: Resultados clasificación ataque DoSGoldeneye



Fuente: Elaboración propia

Para el ataque DoSGoldeneye se observa que los 3 modelos clasifican en mayor medida sus tramas, y su cantidad de solicitudes se puede comparar con la del ataque DDoS, observado en la figura 6-16.

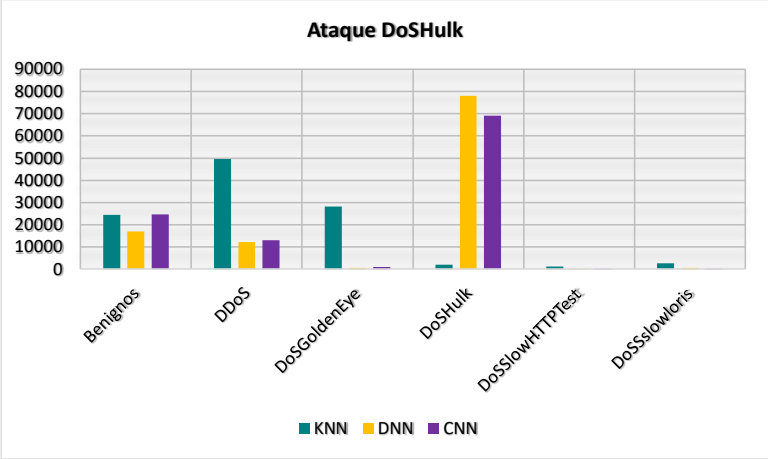
Figura 6-18: Resultados clasificación ataque DoSSlowHTTPTest



Fuente: Elaboración propia

Tal como se observa en la figura 6-18, el ataque DoSSlowHTTPTTest realiza un número de peticiones similares a las definidas en el ataque DoSSlowloris, sin embargo, puede ser considerado un ataque con mayor contundencia dada la cantidad de solicitudes registradas en el mismo período de tiempo en comparación a DoSSlowloris. Los modelos que mejor comportamiento arrojaron fueron CNN y DNN.

Figura 6-19: Resultados clasificación ataque DoSHulk



Fuente: Elaboración propia

Tal como se pudo observar en la figura 6-19, el ataque DoSHulk fue el que mayor cantidad de tráfico generó en el tiempo determinado para la prueba en comparación a los demás ataques. Este comportamiento puede generar capturas de tráfico en formato .pcap que superen las 2 gigas en pocos minutos. Tanto la red neuronal profunda y la red neuronal convolucional clasificaron principalmente el ataque adecuadamente.

6.5 Resultados Prueba 2

A continuación, se observan los resultados obtenidos según la clasificación de los modelos desde la tabla 6-8 hasta la tabla 6-12.

Tabla 6-8: Resultados ataque DoSSlowloris

Ataque DoSSlowloris							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	118	1	162	0	150	11	442
DNN	40	2	1	0	0	399	442
CNN	40	0	1	0	1	400	442

Fuente: Elaboración propia

Tabla 6-9: Resultados ataque DDoS

Ataque DDoS							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	1887	0	56	0	6	4233	6182
DNN	72	5984	1	0	96	29	6182
CNN	48	6112	20	1	1	0	6182

Fuente: Elaboración propia

Tabla 6-10: Resultados ataque DoSGoldeneye

Ataque DoSGoldeneye							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	1904	0	5635	28	2791	251	10609
DNN	273	2	7213	7	2292	822	10609
CNN	390	22	7240	6	409	2542	10609

Fuente: Elaboración propia

Tabla 6-11: Resultados ataque DoSSlowHTTPTest

Ataque DoSSlowHTTPTest							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	737	0	176	0	60	64	1037
DNN	17	1	1	1	1008	9	1037
CNN	269	2	3	0	753	10	1037

Fuente: Elaboración propia

Tabla 6-12: Resultados ataque DoSHulk

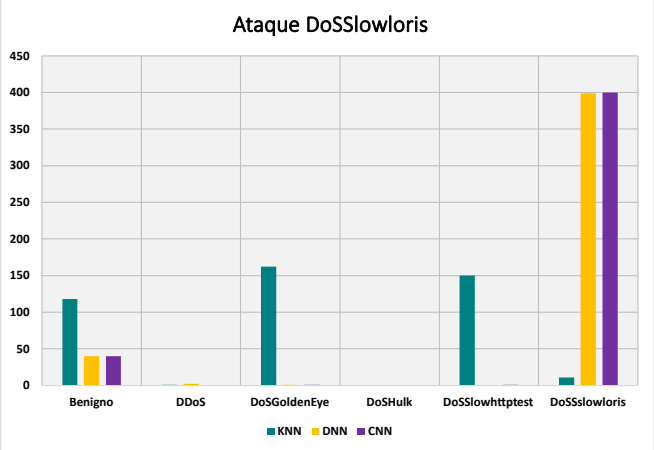
Ataque DoSHulk							
Técnica de clasificación	**Benignos**	**DDoS**	**DoSGoldenEye**	**DoSHulk**	**DoSSlowhttptest**	**DoSSslowloris**	**Total**
KNN	5826	13551	6284	1036	403	731	27831
DNN	1522	458	131	25386	0	334	27831
CNN	2577	886	231	24122	0	15	27831

Fuente: Elaboración propia

A partir de los datos obtenidos de la prueba 2, se representan en las figuras 6-20 hasta 6-20 el comportamiento de los modelos en esta prueba.

En la figura 6-20 se observa que a pesar de que la prueba 2 fue ejecutada en condiciones diferentes con respecto a la prueba 1, el comportamiento del ataque DoSSlowloris se mantiene con relación a lo observado en la figura 6-15. Los clasificadores CNN y DNN pudieron categorizar significativamente el ataque, no obstante, se observa que el tráfico benigno generado por los usuarios puede tener cierta afectación en la clasificación, dado los dominios de tráfico *broadcast* generados por el switch.

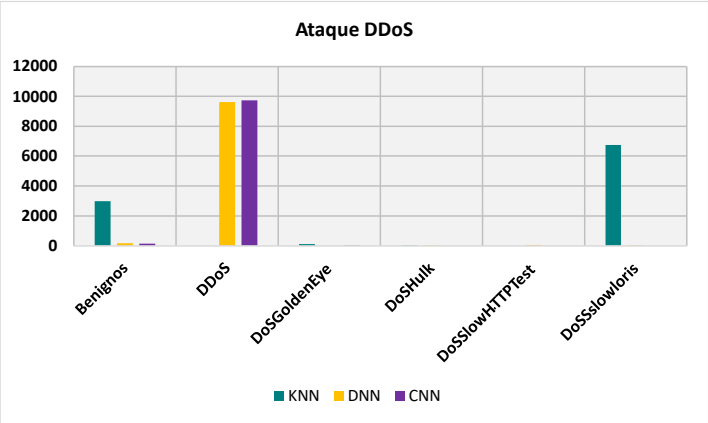
Figura 6-20: Resultados clasificación ataque DoSSlowloris



Fuente: Elaboración propia

Con respecto a la figura 6-21, los modelos DNN y CNN pueden clasificar adecuadamente el tráfico generado por la herramienta HOIC, la cual define ataques de tipo DDoS. Con relación a la prueba 1, la cantidad de peticiones se ve disminuida, lo cual puede explicarse dados los tiempos definidos en cada prueba.

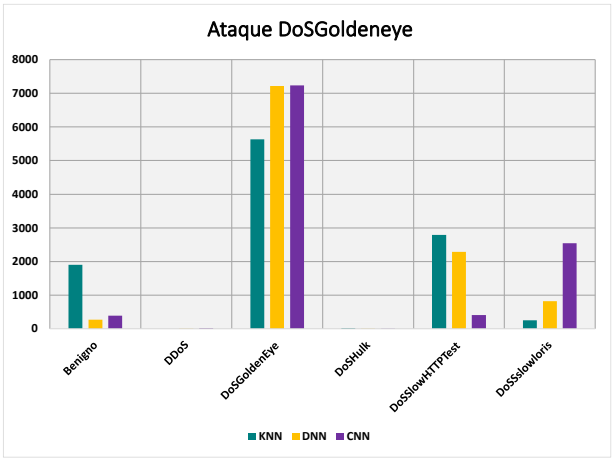
Figura 6-21: Resultados clasificación ataque DDoS



Fuente: Elaboración propia

Tal como se observa en la figura 6-22, los 3 clasificadores determinaron en su mayoría que las peticiones eran correspondientes con el ataque DoSGoldeneye, tal como ocurrió en la prueba 1 (ver figura 6-17). Este tipo de ataque genera una gran cantidad de solicitudes al equipo víctima, en comparación con DoSSlowloris y DoSHTTPTest.

Figura 6-22: Resultados clasificación ataque DoSGoldeneye

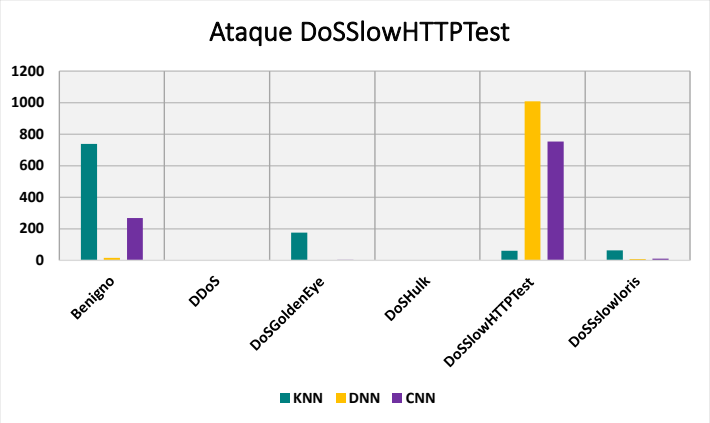


Fuente: Elaboración propia

En la figura 6-23, se observa que el ataque DoSSlowHTTPTest no se caracteriza por un gran envío de paquetes, al igual que ocurre en la prueba 1 (ver figura 6-18), sin embargo, el modelo DNN clasificó adecuadamente el tráfico de red capturado. Con respecto al

modelo CNN, su predicción fue acertada en una menor proporción con respecto a los resultados de DNN.

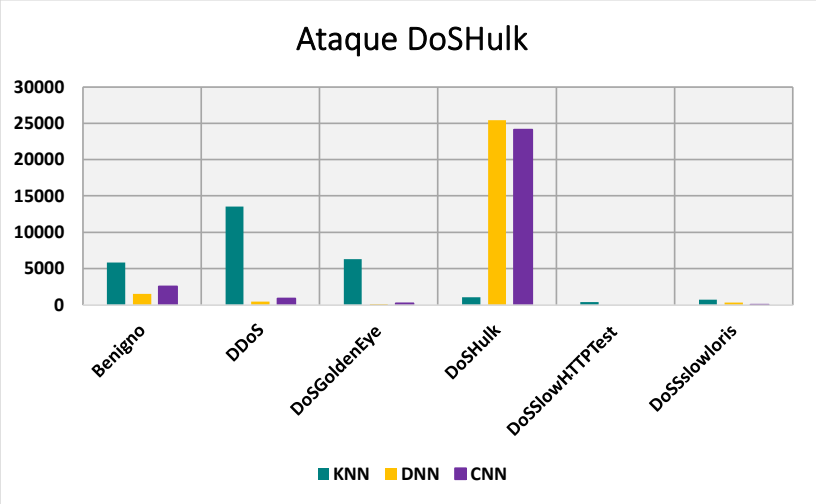
Figura 6-23: Resultados clasificación ataque DoSSlowHTTPTest



Fuente: Elaboración propia

El ataque DoSHulk se consolidó como el ataque que mayor cantidad de tráfico generó, tal como se puede ver en la figura 6-24. Los clasificadores DNN y CNN pudieron realizar predicciones acertadas con relación al ataque, sin embargo, se presentó una tendencia a la clasificación como registros benignos, tal como ocurrió en la prueba 1 (ver figura 6-19).

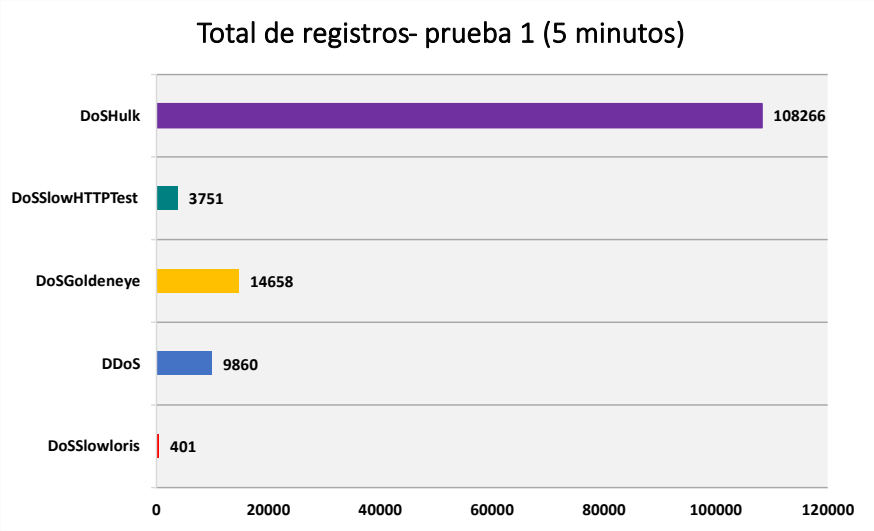
Figura 6-24: Resultados clasificación ataque DoSHulk



Fuente: Elaboración propia

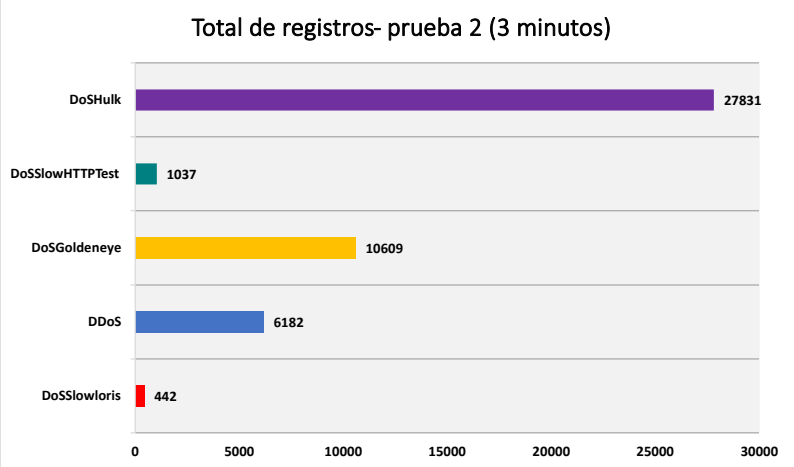
A continuación, se definen 2 gráficas para observar la cantidad de tramas enviadas por cada ataque.

Figura 6-25: Cantidad de tráfico generado durante prueba 1



Fuente: Elaboración propia

Figura 6-26: Cantidad de tráfico generado durante prueba 2



Fuente: Elaboración propia

Tal como se observa en las figuras 6-25 y 6-26, los ataques de tipo *Slow* se caracterizaron en ambas pruebas por su envío de datos moderado, seguidos del ataque DDoS. El ataque DoSGoldeneye generó una cantidad de peticiones considerables, no obstante, DoSHulk demostró ser el más efectivo en cuanto a envío de paquetes, dado que en la prueba 1 se obtuvo un archivo .pcap de 2.1 gigas y para la prueba 2 alcanzó 1.4 gigas. En la tabla 6-

13 se encuentran los enlaces para acceder a los archivos obtenidos de las pruebas realizadas.

Tabla 6-13: Enlaces para archivos de pruebas para validación

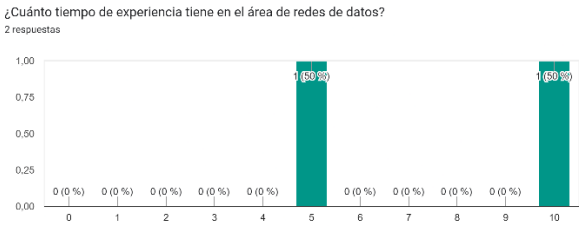
Prueba	Enlace
#1	https://drive.google.com/drive/folders/1kpXlS5bvEc2dO-FBTlbRd6RNYeeR80XK?usp=share_link
#2	https://drive.google.com/drive/folders/1LIrRY3H0f3jiYr0OSwUO4R0OE1FIfQeV?usp=share_link

Fuente: Elaboración propia.

6.6 Validación con Profesionales Expertos

De acuerdo con el enfoque definido en esta tesis de maestría, se estableció un escenario de validación con dos profesionales expertos en el área de redes de cómputo: uno perteneciente a la empresa Confa y el segundo de la Universidad Nacional de Colombia. La ingeniera analizó el sistema en búsqueda de elementos que podrían ser de interés de acuerdo con su rol. Durante la revisión del sistema se verificó el funcionamiento del prototipo local y servicio en la nube. Posteriormente, los expertos respondieron una breve encuesta. A continuación, se exponen las apreciaciones de los profesionales por medio de de un formulario de Google (figuras 6-27 a 6-34). En las figuras 6-31 a 6-34, la opinión está representada por valores entre 1 y 10, siendo 1 la puntuación más baja y 10 la más alta.

Figura 6-27: Tiempo de Experiencia en Redes de Datos



Fuente: Elaboración propia.

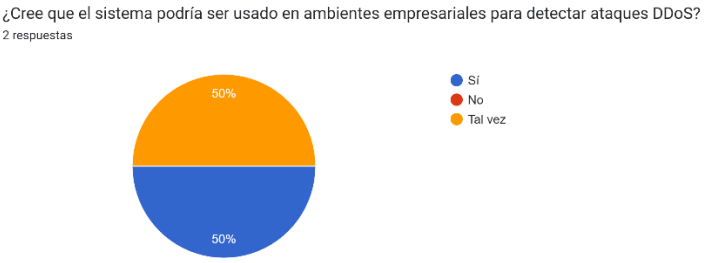
Figura 6-28: Experiencia de la experta en Inteligencia Artificial

Fuente: Elaboración propia.

Figura 6-29: Necesidad de Conocimiento en Inteligencia Artificial

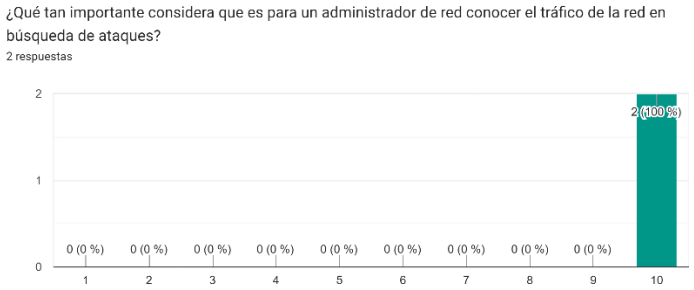
Fuente: Elaboración propia.

Figura 6-30: Uso del IDS en Ambientes Reales



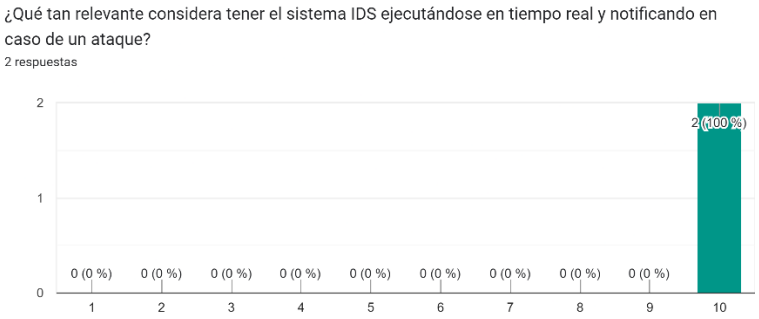
Fuente: Elaboración propia.

Figura 6-31: Importancia para Reconocer Ataques



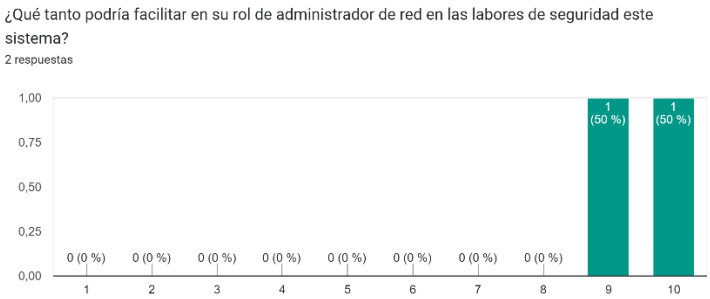
Fuente: Elaboración propia.

Figura 6-32: Importancia del Análisis en Tiempo Real



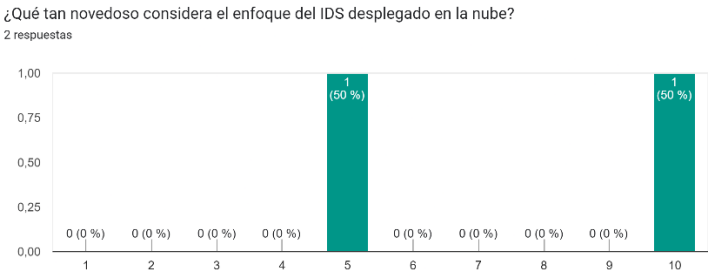
Fuente: Elaboración propia.

Figura 6-33: Aporte del Sistema en Términos Laborales



Fuente: Elaboración propia.

Figura 6-34: Novedad en el Enfoque de IDS



Fuente: Elaboración propia.

Enlace con respuestas: https://docs.google.com/spreadsheets/d/1CK-af1PWCR1CiGptudfhIqgntunpYBgI1r0DglV4dxk/edit?usp=sharing

Las apreciaciones de la ingeniera indican la necesidad de implementar mecanismos que aumenten la tasa de detección de Ataques de Denegación de Servicios Distribuidos, y en particular por medio de sistemas que generen reportes dinámicos y funcionamiento en tiempo real. Como aspectos a mejorar, la experta recomienda incluir mayores detalles sobre el comportamiento de las tramas, de tal modo, que sea posible identificar el protocolo, dirección IP y el tamaño de los paquetes que están asociados al ataque.

Adicionalmente, resalta la importancia de notificar al usuario con la cantidad exacta de registros definidos por cada ataque y la viabilidad de implementación en una red de cómputo a nivel empresarial.

6.7 Conclusiones del Capítulo

En este capítulo se realizaron pruebas para analizar el comportamiento del prototipo IDS soportado en técnicas de aprendizaje supervisado para la detección de ataques de tipo DoS/DDoS orientado a servicios en la nube.

Las pruebas realizadas permiten confirmar que es viable construir sistemas de detección de intrusos soportados en técnicas de aprendizaje supervisado orientado a servicios en la nube para la detección de ataques de denegación de servicios distribuidos, dado su buen

comportamiento al momento de analizar y detectar ataques DoS/DDoS en el tráfico generado en una red de datos.

La validación con la profesional experta indica que es viable considerar un despliegue del sistema IDS en ambientes empresariales y recomienda la implementación de características que aumenten en nivel de detalle en tráfico de red.

7 Capítulo 7: Consideraciones finales

7.1 Conclusiones

La investigación llevada a cabo permitió la construcción de un prototipo de sistema de detección de intrusos (IDS) bajo una arquitectura orientada a servicios. Este prototipo cuenta con dos enfoques de funcionamiento: manual y automático.

El enfoque manual requiere de un usuario para iniciar la captura del tráfico, conversión a formato CSV y selección de técnicas para determinar la clasificación. El enfoque automático solicita al usuario la parametrización para enviar una notificación vía SMS en caso de la detección de un ataque, dado que el prototipo estará en ejecución hasta la detección de un ataque DoS/DDoS.

Además, ofrece tres servicios específicos: captura de tráfico de red mediante las interfaces dinámicas del equipo donde se despliegue, conversión de las tramas del tráfico de red en un archivo en formato CVS del dataset CICIDS2017, y clasificación del tráfico por medio técnicas de aprendizaje supervisado

Como aspectos relevantes que esta Tesis puede aportar al estado del arte se define la posibilidad de agregar nuevos servicios al prototipo dada su escalabilidad. Adicionalmente, la posibilidad de integrar la detección de nuevos tipos de ataques si se define el entrenamiento necesario. Por último, se resalta la validación del prototipo por medio de una subred en la cual se presentó tráfico real generado por usuarios, lo cual permitió conocer las dificultades que pueden generarse al momento de validar modelos entrenados con un conjunto de datos limpio y sin ningún tipo de ruido.

La presente Tesis se enfoca en aportar al estado del arte en cuanto a la construcción de un prototipo de sistema de detección de intrusos (IDS) bajo una arquitectura orientada a servicios. Se destacan como aspectos relevantes la validación del prototipo a través de una subred en la cual se presentó tráfico real generado por usuarios.

7.2 Cumplimiento de los objetivos

Los objetivos propuestos para la presente Tesis se alcanzaron así:

Objetivo 1: Determinar la(s) técnica(s) de aprendizaje supervisado relevantes en detección de ataques DDoS.

Cumplimiento: Se determinaron las técnicas de aprendizaje supervisado relevantes en la detección de ataques de denegación de servicio distribuidos por medio del estudio paralelo descrito en el capítulo 4.

Objetivo 2: Diseñar la arquitectura del prototipo orientado a servicios.

Cumplimiento: Se definió el diseño de una arquitectura orientada a servicios compuesta de tres servicios: captura de tráfico, conversión de tramas y clasificación. Adicionalmente, se determinaron dos enfoques: enfoque manual y enfoque automático. El análisis detallado se encuentra explícito en el capítulo 5.

Objetivo 3: Implementar el prototipo computacional de un IDS basado en aprendizaje supervisado en arquitectura orientada a servicios en la Nube.

Cumplimiento: La implementación del prototipo fue desarrollada bajo la arquitectura orientada a servicios en tecnologías libres y para ambientes Linux. En el capítulo 5 se consolidó toda la información del proceso de desarrollo.

Objetivo 4: Verificación y validación de la implementación orientada a servicios del IDS con un dataset reconocido por la comunidad científica.

Cumplimiento: La validación del prototipo se realizó por medio de pruebas realizadas en las instalaciones de la Universidad Nacional de Colombia. En el capítulo 6 se encuentran los resultados obtenidos de cada una de las mediciones.

7.3 Trabajo futuro

Como trabajo futuro, se espera ampliar las funcionalidades del prototipo de sistema de detección de intrusos (IDS) incluyendo acciones preventivas para dar por terminado un ataque detectado. Asimismo, se buscará mejorar la precisión de los modelos de clasificación mediante la incorporación de nuevos patrones de detección de diferentes tipos de ataques. De igual manera, se plantea la actualización de algunos servicios a implementaciones REST que permita la carga archivos en formato .pcap y se realice la conversión y sea posible obtener un análisis de tráfico capturado. Adicionalmente, se

espera implementar un módulo con información relacionada con DoS y DDoS, de modo tal, que se genere que sea posible fomentar el aprendizaje sobre este tipo de ataques

8 Bibliografía

Abughazleh, A., Almiani, M., Magableh, B., & Razaque, A. (2019). Intelligent intrusion detection using radial basis function neural network. *2019 6th International Conference on Software Defined Systems, SDS 2019*, 200–208. https://doi.org/10.1109/SDS.2019.8768575

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021a). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, *32*(1). https://doi.org/10.1002/ett.4150

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021b). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, *32*(1). https://doi.org/10.1002/ett.4150

Almanza J., A. R. (2019). XIX Encuesta Nacional de Seguridad Informática. *Revista SISTEMAS*, *151*, 12–41. https://doi.org/10.29236/sistemas.n151a3

Almseidin, M., & Kovacs, S. (2019). Intrusion detection mechanism using fuzzy rule interpolation. *ArXiv*.

Alonso, A., Gomez, R., Juan, E., Jaimes, C., Francisco, U., & Santander, D. P. (2018). *Recibido: 13 de septiembre de 2017 Aceptado: 8 de diciembre de 2017*.

Alzubi, O. A., Alzubi, J. A., Alazab, M., Alrabea, A., Awajan, A., & Qiqieh, I. (2022). Optimized Machine Learning-Based Intrusion Detection System for Fog and Edge Computing Environment. *Electronics (Switzerland)*, *11*(19). https://doi.org/10.3390/electronics11193007

Andrade Carrera, H., Sinche Maita, S., & Hidalgo Lascano, P. (2021). Modelo para detectar el uso correcto de mascarillas en tiempo real utilizando redes neuronales convolucionales. *Revista de Investigación En Tecnologías de La Información*, *9*(17), 111–120. https://doi.org/10.36825/riti.09.17.011

Arango Serna, M. D., Londoño Salazar, J. E., & Zapata Cortes, J. A. (2010). Arquitectura orientada a servicios en el contexto de la arquitectura empresaria. *Avances En Sistemas e Informática*, *7*, 15–88.

Asharf, J., Moustafa, N., Khurshid, H., Debie, E., Haider, W., & Wahab, A. (2020). A review of intrusion detection systems using machine and deep learning in internet of things: Challenges, solutions and future directions. In *Electronics (Switzerland)* (Vol. 9, Issue 7). MDPI AG. https://doi.org/10.3390/electronics9071177

Azahari Mohd, Y. M., Fakariah Hani Mohd, A., & Mohamad Yusof, D. (2018). Detection and Defense Algorithms of Different Types of DDoS Attacks. *International Journal of Engineering and Technology*, *9*(5), 410–444. https://doi.org/10.7763/ijet.2017.v9.1008

Balaji, S., Salih, A., & Al-Atroshi, C. (2018). Adaptability of SOA in IoT Services – An Empirical Survey. *International Journal of Computer Applications*, *182*(31), 25–28. https://doi.org/10.5120/ijca2018918249

Bautista, J., Tutores, R., & Manzano, L. G. (2018). *Ataques DDoS con IoT, Análisis y Prevención de Riesgos*. https://e-archivo.uc3m.es/handle/10016/29630

Bebortta, S., & Singh, S. K. (2021). An Adaptive Machine Learning-based Threat Detection Framework for Industrial Communication Networks. *2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)*. https://doi.org/10.1109/CSNT.2021.91

Beckmann, M., Ebecken, N. F. F., & Pires de Lima, B. S. L. (2015). A KNN Undersampling Approach for Data Balancing. *Journal of Intelligent Learning Systems and Applications*, *07*(04), 104–116. https://doi.org/10.4236/jilsa.2015.74010

Beechey, M., Lambotharan, S., & Kyriakopoulos, K. G. (2023). Evidential classification for defending against adversarial attacks on network traffic. *Information Fusion*, *92*, 115–126. https://doi.org/10.1016/j.inffus.2022.11.024

Benza¨ıdbenza¨ıd, C., Boukhalfa, M., & Taleb, T. (2020). Robust Self-Protection Against Application-Layer (D)DoS Attacks in SDN Environment. *IEEE Wireless Communications and Networking Conference (WCNC)*. https://github.com/grafov/hulk

Bhardwaj, A., Mangat, V., & Vig, R. (2020). Hyperband Tuned Deep Neural Network With Well Posed Stacked Sparse AutoEncoder for Detection of DDoS Attacks in Cloud. *IEEE Access*, *8*, 181916–181929. https://doi.org/10.1109/access.2020.3028690

Black, S., & Kim, Y. (2022). An Overview on Detection and Prevention of Application Layer DDoS Attacks. *2022 IEEE 12th Annual Computing and Communication Workshop and Conference, CCWC 2022*, 791–800. https://doi.org/10.1109/CCWC54503.2022.9720741

Bogdanoski, M. (2013). *Analysis of the SYN Flood DoS Attack*. *June*, 1–11. https://doi.org/10.5815/ijcnis.2013.08.01

Bravo, S., & Mauricio, D. (2019). Systematic review of aspects of DDoS attacks detection. *Indonesian Journal of Electrical Engineering and Computer Science*, *14*(1), 155. https://doi.org/10.11591/ijeecs.v14.i1.pp155-168

Caballero, J. S., De, L., Sánchez, P., Jorge, P., & López De Vergara, E. (2019). *Estudio de detección de ciberataques en Internet mediante algoritmos de clasificación de parámetros de tráfico*.

Cano, J. J. (2012). *Seguridad de la información y privacidad: dos conceptos convergentes*. http://acis.org.co/revistasistemas/index.php/ediciones-revista-sistemas/edicion-no-123/item/100-seguridad-de-la-informaci﷿n-y-privacidad-dos-conceptos-convergentes

Carlin, A., Hammoudeh, M., & Aldabbas, O. (2015). Defence for Distributed Denial of Service Attacks in Cloud Computing. *Procedia Computer Science*, *73*, 490–497. https://doi.org/10.1016/j.procs.2015.12.037

Čelesová, B., Val'ko, J., Grežo, R., & Helebrandt, P. (2019). Enhancing security of SDN focusing on control plane and data plane. *7th International Symposium on Digital Forensics and Security, ISDFS 2019*, 1–6. https://doi.org/10.1109/ISDFS.2019.8757542

Chang, V., Golightly, L., Modesti, P., Xu, Q. A., Doan, L. M. T., Hall, K., Boddu, S., & Kobusińska, A. (2022). A Survey on Intrusion Detection Systems for Fog and Cloud Computing. *Future Internet*, *14*(3). https://doi.org/10.3390/fi14030089

Chaudhary, A., & Shrimal, G. (2019). Intrusion Detection System Based on Genetic Algorithm for Detection of Distribution Denial of Service Attacks in MANETs. *SSRN Electronic Journal*, 370–377. https://doi.org/10.2139/ssrn.3351807

Cisar, P., & Pinter, R. (2019). *Journal of Applied Technical and Educational Sciences jATES Some ethical hacking possibilities in Kali Linux environment*. *9*(4), 129–149. https://doi.org/10.24368/jates.v9i4.139

Correa Wachter, J. F., & Henao Villas, C. F. (2021). Análisis del aporte del aprendizaje de máquinas a la seguridad de la información. *Ingente Americana*, *1*(1), 9–20. https://doi.org/10.21803/ingecana.1.1.407

Croft, R., Xie, Y., & Babar, M. A. (2022). Data Preparation for Software Vulnerability Prediction: A Systematic Literature Review. *IEEE Transactions on Software Engineering*. https://doi.org/10.1109/TSE.2022.3171202

Disha, R. A., & Waheed, S. (2022). Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest

(GIWRF) feature selection technique. *Cybersecurity*, *5*(1). https://doi.org/10.1186/s42400-021-00103-8

Dong, S., Su, H., & Liu, Y. (2022). A-CAVE: Network abnormal traffic detection algorithm based on variational autoencoder. *ICT Express*. https://doi.org/10.1016/j.icte.2022.11.006

Doriguzzi-Corin, R., Millar, S., Scott-Hayward, S., Martínez-del-Rincón, J., & Siracusa, D. (2020). LUCID: A practical, lightweight deep learning solution for DDoS attack detection. *ArXiv*, *17*(2), 876–889.

Farhana, N., Firdaus, A., Darmawan, M. F., & Ab Razak, M. F. (2022). Evaluation of Boruta algorithm in DDoS detection. *Egyptian Informatics Journal*. https://doi.org/10.1016/j.eij.2022.10.005

Figueroa, J. A., Rodríguez-Andrade, R. F., Bone-Obando, C. C., & Saltos-Gómez, J. A. (2018). La seguridad informática y la seguridad de la información. *Polo Del Conocimiento*, *2*(12), 145. https://doi.org/10.23857/pc.v2i12.420

Gangula, R., Mohan, V. M., & Kumar, R. (2022). A comprehence study of DDoS attack detecting algorithm using GRU-BWFA classifier. *Measurement: Sensors*, *24*. https://doi.org/10.1016/j.measen.2022.100570

Gholizadeh, S. (2022). Top Popular Python Libraries in Research. In *Journal of Robotics and Automation Research* (Vol. 3, Issue 2). www.opastonline.com

Hadeel S. Obaid. (2020). Denial of Service Attacks: Tools and Categories. *International Journal of Engineering Research And*, *V9*(03), 631–636. https://doi.org/10.17577/ijertv9is030289

Haider, S., Akhunzada, A., Mustafa, I., Patel, T. B., Fernandez, A., Choo, K. K. R., & Iqbal, J. (2020). A Deep CNN Ensemble Framework for Efficient DDoS Attack Detection in Software Defined Networks. *IEEE Access*, *8*, 53972–53983. https://doi.org/10.1109/ACCESS.2020.2976908

Harshita, H. (2017). Detection and Prevention of ICMP Flood DDOS Attack. *International Journal of New Technology and Research*, *3*(3), 263333.

Hasnain, M., Pasha, M. F., Ghani, I., Imran, M., Alzahrani, M. Y., & Budiarto, R. (2020). Evaluating Trust Prediction and Confusion Matrix Measures for Web Services Ranking. *IEEE Access*, *8*, 90847–90861. https://doi.org/10.1109/ACCESS.2020.2994222

Humayun, M., Niazi, M., Jhanjhi, N., Alshayeb, M., & Mahmood, S. (2020). Cyber Security Threats and Vulnerabilities: A Systematic Mapping Study. *Arabian Journal for*

Science and Engineering, *45*(4), 3171–3189. https://doi.org/10.1007/s13369-019-04319-2

Iram, A., Zahrah, A., Faheem, M., & Alwi M, B. (2020). *A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset*.

Isaza, G., & Ramírez, F. (2023). *Prototipo de Red Neuronal Profunda Aplicada en Ciberseguridad* [Tesis]. Universidad de Caldas.

Issa, A., Albayrak, Z., & Sardar, A. (2023). DDoS Attack Intrusion Detection System Based on Hybridization of CNN and LSTM. *Acta Polytechnica Hungarica*, *20*(2), 105–123. https://doi.org/10.12700/APH.20.3.2023.3.6

Krishnaveni, S., Sivamohan, S., Sridhar, S. S., & Prabakaran, S. (2021). Efficient feature selection and classification through ensemble method for network intrusion detection on cloud computing. *Cluster Computing*, *24*(3), 1761–1779. https://doi.org/10.1007/s10586-020-03222-y

Kumar, V. (2014). Feature Selection: A literature Review. *The Smart Computing Review*, *4*(3). https://doi.org/10.6029/smartcr.2014.03.007

Lachnit, S., Gallenmüller, S., Scholz, D., & Stubbe, H. (2021). xdpcap: XDP Packet Capture. *Network Architectures and Services*, 41–44. https://doi.org/10.2313/NET-2021-05-1_09

Lafram, I., Berbiche, N., & el Alami, J. (2019). Artificial Neural Networks Optimized with Unsupervised Clustering for IDS Classification. *ICSSD 2019 - International Conference on Smart Systems and Data Science*. https://doi.org/10.1109/ICSSD47982.2019.9002827

Layme Fernández, C., Manuel, J., Canaza, S., Jose, D., Ugarte, P., Yoset, J., & Quispe, L. (2022). Application of decision trees in the identification of fraudulent websites. *Revista Innovación y Software*, *3*(1).

Le, T. T. H., Kim, Y., & Kim, H. (2019). Network intrusion detection based on novel feature selection model and various recurrent neural networks. *Applied Sciences (Switzerland)*, *9*(7). https://doi.org/10.3390/app9071392

Lee, K. J. (2021). Architecture of neural processing unit for deep neural networks. In *Advances in Computers* (Vol. 122, pp. 217–245). Academic Press Inc. https://doi.org/10.1016/bs.adcom.2020.11.001

Lee, T. H., Chang, L. H., & Syu, C. W. (2020). Deep learning enabled intrusion detection and prevention system over SDN networks. *2020 IEEE International Conference on Communications Workshops, ICC Workshops 2020 - Proceedings*. https://doi.org/10.1109/ICCWorkshops49005.2020.9145085

Leevy, J. L., Hancock, J., Zuech, R., & Khoshgoftaar, T. M. (2021). Detecting cybersecurity attacks across different network features and learners. *Journal of Big Data*, *8*(1). https://doi.org/10.1186/s40537-021-00426-w

Lublinsky, B. (2007). Defining SOA as an architectural style: Align your business model with technology. *IBM DeveloperWorks Site*.

Maldonado, J. (2018). *Avances en Sistemas de Detecci ´ on de Intrusos con on Sistem ´ atica de la Literatura*. 978–980.

Manso, P., Moura, J., & Serrão, C. (2019). SDN-based intrusion detection system for early detection and mitigation of DDoS attacks. *Information (Switzerland)*, *10*(3), 1–17. https://doi.org/10.3390/info10030106

Merchán, G., & César, E. (2022). *Transhumanismo y consciencia fenoménica Transhumanism and phenomenal consciousness*. 109–126.

Mishra, N., & Pandya, S. (2021). Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review. In *IEEE Access* (Vol. 9, pp. 59353–59377). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2021.3073408

Mohammadpour, L., Ling, T. C., Liew, C. S., & Aryanfar, A. (2022). A Survey of CNN-Based Network Intrusion Detection. In *Applied Sciences (Switzerland)* (Vol. 12, Issue 16). MDPI. https://doi.org/10.3390/app12168162

Montes-Gil, J. A., Isaza-Cadavid, G., & Duque-Méndez, N. D. (2023). Efecto de la selección de atributos en el desempeño de un IDS basado en machine learning para detección de intrusos en ataques DDoS. *South Florida Journal of Development*, *4*(2), 918–928. https://doi.org/10.46932/sfjdv4n2-023

Moukafih, N., Orhanou, G., & el Hajji, S. (2020). Neural Network-Based Voting System with High Capacity and Low Computation for Intrusion Detection in SIEM/IDS Systems. *Security and Communication Networks*, *2020*. https://doi.org/10.1155/2020/3512737

Moukhaf, M., El Yassini, K., Bri, S., & Oufaska, K. (2019). *Building an e-learning recommender system using Association Rules techniques and R environment* (Vol. 3, Issue 2). Springer International Publishing. https://doi.org/10.1007/978-3-030-11928-7

Muraleedharan, N., & Janet, B. (2021). A deep learning based HTTP slow DoS classification approach using flow data. *ICT Express*, *7*(2), 210–214. https://doi.org/10.1016/j.icte.2020.08.005

Niknejad, N., Ismail, W., Ghani, I., Nazari, B., Bahari, M., & Hussin, A. R. B. C. (2020). Understanding Service-Oriented Architecture (SOA): A systematic literature review and directions for further investigation. *Information Systems*, *91*. https://doi.org/10.1016/j.is.2020.101491

Ocampo, C. A., Viviana, Y., Bermúdez, C., & Solarte Martínez, G. R. (2017). Sistema de detección de intrusos en redes corporativas Intrusion Detection System in Corporate Networks. *Scientia et Technica Año XXII*, *22*(1), 122–170.

Panigrahi, R., & Borah, S. (2018). A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems Analysis of Selected Clustering Algorithms Used in Intrusion Detection Systems View project IEEE International Conference on Advanced Computational and Communication Paradigms (ICACCP-2017) View project A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. In *Article in International Journal of Engineering & Technology* (Vol. 7, Issue 3). https://www.researchgate.net/publication/329045441

Pantoja, N. D., Donado, A. S., & Villalba, M. K. (2019). Selección de indicadores para la implementación de un IDS en PYMES. *Risti*, 777–786.

Pawlicki, M., Choraś, M., & Kozik, R. (2020). Defending network intrusion detection systems against adversarial evasion attacks. *Future Generation Computer Systems*, *110*, 148–154. https://doi.org/10.1016/j.future.2020.04.013

Phan, T. V., Nguyen, T. G., Dao, N. N., Huong, T. T., Thanh, N. H., & Bauschert, T. (2020). DeepGuard: Efficient Anomaly Detection in SDN with Fine-Grained Traffic Flow Monitoring. *IEEE Transactions on Network and Service Management*, *17*(3), 1349–1362. https://doi.org/10.1109/TNSM.2020.3004415

Preuveneers, D., Rimmer, V., Tsingenopoulos, I., Spooren, J., Joosen, W., & Ilie-Zudor, E. (2018). Chained anomaly detection models for federated learning: An intrusion detection case study. *Applied Sciences (Switzerland)*, *8*(12). https://doi.org/10.3390/app8122663

Puder, A., Rumez, M., Grimm, D., & Sax, E. (2022). Generic Patterns for Intrusion Detection Systems in Service-Oriented Automotive and Medical Architectures. *Journal of Cybersecurity and Privacy*, *2*(3), 731–749. https://doi.org/10.3390/jcp2030037

Quiroz, & Valencia, D. (2017). Seguridad en informática: consideraciones. *Dominio de Las Ciencias*, *3*(3), 676–688.

Quirumbay Yagual, D. I., Castillo Yagual, C., & Coronel Suárez, I. (2022). Una revisión del Aprendizaje profundo aplicado a la ciberseguridad. *Revista Científica y Tecnológica UPSE*, *9*(1), 57–65. https://doi.org/10.26423/rctu.v9i1.671

Rojas, C., Sebastian, B., Rodríguez, C., Uriel, C., Osorio, E., Javier, D., Tatiana, Y., Universitaria, F., & Gil, D. S. (2020). *Redes neuronales artificiales y estado del arte aplicado en la ciberseguridad State of the art artificial networks applied to cybersegurity*. 58–63.

Roopak, M., Tian, G. Y., & Chambers, J. (2020). An Intrusion Detection System Against DDoS Attacks in IoT Networks. *2020 10th Annual Computing and Communication Workshop and Conference, CCWC 2020*, 562–567. https://doi.org/10.1109/CCWC47524.2020.9031206

Rt', K., Selvi', S. T., & Govindarajan2, K. (2014). *DDoS Detection and Analysis in SDN-based Environment Using Support Vector Machine Classifier*.

S, S. A., Ramesh Babu, D. R., & Venkatesan, S. (2019). Twilio Integration with Dialogflow for Effective Communication. *Journal of Web Development and Web Designing*, *4*(2). https://doi.org/10.5281/zenodo.3251169

Sallam, A. A., Kabir, M. N., Alginahi, Y. M., Jamal, A., & Esmeel, T. K. (2020). IDS for Improving DDoS Attack Recognition Based on Attack Profiles and Network Traffic Features. *Proceedings - 2020 16th IEEE International Colloquium on Signal Processing and Its Applications, CSPA 2020*, *Cspa*, 255–260. https://doi.org/10.1109/CSPA48992.2020.9068679

Sampedro, C. R., Machuca Vivar, S. A., Palma Rivera, D. P., & Carrera Calderón, F. A. (2019). Percepción de seguridad de la información en las pequeñas y medianas empresas en santo domingo. *Investigacion Operacional*, *40*(3), 421–428.

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSP 2018 - Proceedings of the 4th International Conference on Information Systems Security and Privacy*, *2018-January*, 108–116. https://doi.org/10.5220/0006639801080116

Sharma, N. V., & Yadav, N. S. (2021). An optimal intrusion detection system using recursive feature elimination and ensemble of classifiers. *Microprocessors and Microsystems*, *85*. https://doi.org/10.1016/j.micpro.2021.104293

Shi, Z., Li, J., & Wu, C. (2019). DeepDDoS: Online DDoS attack detection. *2019 IEEE Global Communications Conference, GLOBECOM 2019 - Proceedings*, 0–5. https://doi.org/10.1109/GLOBECOM38437.2019.9013186

Singh, A., Akash, R., & Gokul Rajan, V. (2022). Flower Classifier Web App Using Ml & Flask Web Framework. *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering, ICACITE 2022*, 974–977. https://doi.org/10.1109/ICACITE53722.2022.9823577

Snigdho, M. A., Chowdhury, S., & Jahan, N. (2022). Real-Time DDoS Attack Prediction using Supervised Algorithms and CNN. *7th International Conference on Communication and Electronics Systems, ICCES 2022 - Proceedings*, 1342–1348. https://doi.org/10.1109/ICCES54183.2022.9835977

Solanki, S. D., & Solanki, A. D. (2020). Identification of DOS Attack by implementing SYN Flood Attack and considering CPU Load Analysis. *International Research Journal on Advanced Science Hub*, *2*(8), 68–74. https://doi.org/10.47392/irjash.2020.96

Spiekermann, D., & Keller, J. (2021). Unsupervised packet-based anomaly detection in virtual networks. *Computer Networks*, *192*. https://doi.org/10.1016/j.comnet.2021.108017

Susa Velandia, J. S., Carvajal Hoyos, A. C., & Cadena Muñoz, E. (2022). *Modelo de Detección de Ataques DDoS (Distributed Denial of Services), con Base en el Clasificador Decision Tree*. https://repository.udistrital.edu.co/handle/11349/30117

Sze, V., Chen, Y.-H., Yang, T.-J., & Emer, J. (2017). *Efficient Processing of Deep Neural Networks: A Tutorial and Survey*. http://arxiv.org/abs/1703.09039

Taheri, S., Salem, M., & Yuan, J. S. (2018). Leveraging image representation of network traffic data and transfer learning in botnet detection. *Big Data and Cognitive Computing*, *2*(4), 1–16. https://doi.org/10.3390/bdcc2040037

Thakkar, A., & Lohiya, R. (2021). Attack classification using feature selection techniques: a comparative study. *Journal of Ambient Intelligence and Humanized Computing*, *12*(1), 1249–1266. https://doi.org/10.1007/s12652-020-02167-9

Virupakshar, K. B., Asundi, M., Channal, K., Shettar, P., Patil, S., & Narayan, D. G. (2020). Distributed Denial of Service (DDoS) Attacks Detection System for OpenStack-based Private Cloud. *Procedia Computer Science*, *167*(2019), 2297–2307. https://doi.org/10.1016/j.procs.2020.03.282

Wang, J., Chang, X., Wang, Y., Rodríguez, R. J., & Zhang, J. (2021). LSGAN-AT: enhancing malware detector robustness against adversarial examples. *Cybersecurity*, *4*(1). https://doi.org/10.1186/s42400-021-00102-9

Wang, W., Chakraborty, G., & Chakraborty, B. (2021). Predicting the risk of chronic kidney disease (Ckd) using machine learning algorithm. *Applied Sciences (Switzerland)*, *11*(1), 1–17. https://doi.org/10.3390/app11010202

Wu, X., Kumar, V., Ross, Q. J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., Zhou, Z. H., Steinbach, M., Hand, D. J., & Steinberg, D. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, *14*(1), 1–37. https://doi.org/10.1007/s10115-007-0114-2

Xiao, Y., Xing, C., Zhang, T., & Zhao, Z. (2019). An Intrusion Detection Model Based on Feature Reduction and Convolutional Neural Networks. *IEEE Access*, *7*, 42210–42219. https://doi.org/10.1109/ACCESS.2019.2904620