universidad de león

Escuela de Ingenierías

Industrial, Informática y Aeroespacial

MÁSTER UNIVERSITARIO EN INVESTIGACIÓN EN CIBERSEGURIDAD

Trabajo de Fin de Máster

ANÁLISIS EN FLUJO DE DATOS DE DENEGACIÓN DE SERVICIOS

DENIAL OF SERVICE DATA FLOW ANALYSIS

Autor: Fernando Fernández Iglesias
Tutor: Ángel Manuel Guerrero Higueras
Cotutor: Ignacio Samuel Crespo Martínez

(Septiembre, 2023)

UNIVERSIDAD DE LEÓN

Escuela de Ingenierías Industrial, Informática y Aeroespacial

MÁSTER UNIVERSITARIO EN INVESTIGACIÓN EN CIBERSEGURIDAD

Trabajo de Fin de Máster

ALUMNO: Fernando Fernández Iglesias

TUTOR: Ángel Manuel Guerrero Higueras

TÍTULO: Análisis en flujo de datos de denegación de servicios

TITLE: Denial of Service Data Flow Analysis

CONVOCATORIA: Septiembre, 2023

RESUMEN:

En el presente trabajo, se aborda el análisis de las Denegaciones de Servicio (DoS) desde una perspectiva centrada en la detección de estas amenazas mediante el análisis de datos de flujo. Si bien las DoS son detectables en redes de área local con sistemas de detección de intrusiones (IDSs) convencionales, este estudio se enfoca en la posibilidad de detectarlas en redes de área extensa mediante el análisis de datos de flujo generados por routers core de internet.

Para lograr este objetivo, se inicia con una revisión exhaustiva del contexto actual de las DoS, su funcionamiento y los ataques más comunes. Además, se examinan las tendencias actuales en el ámbito del aprendizaje automático aplicado a la detección de tráfico anómalo asociado a las Denegaciones de Servicio.

El análisis de datos de flujo provocados por ataques de DoS se llevó a cabo mediante la implementación de un escenario de denegación de servicio utilizando la herramienta "thorshamer" en "DOROTHEA". Esta herramienta permite la simulación de tráfico benigno y malicioso, la ejecución de ataques y la recopilación de datos resultantes, que se almacenan en un conjunto de datos. A partir de aquí, se aplicaron diversas técnicas de preprocesamiento de datos y algoritmos de aprendizaje automático, evaluando sus indicadores clave de rendimiento (KPIs).

ABSTRACT:

In the present work, we address the analysis of Denial of Service (DoS) attacks from a perspective focused on the detection of these threats through the analysis of flow data. While DoS attacks can be detected in local area networks with conventional Intrusion Detection Systems (IDSs), this study focuses on the possibility of detecting them in wide area networks through the analysis of flow data generated by core internet routers.

To achieve this objective, we begin with a comprehensive review of the current context of DoS attacks, their operation, and the most common attack patterns. Additionally, we examine current trends in the field of machine learning applied to the detection of anomalous traffic associated with Denial of Service incidents.

The analysis of flow data resulting from DoS attacks was conducted by implementing a denial of service scenario using the "thorshamer" tool within "DOROTHEA." This tool allows for the simulation of benign and malicious traffic, the execution of attacks, and the collection of resulting data, which is stored in a dataset. Subsequently, various data preprocessing techniques and machine learning algorithms were applied, evaluating their key performance indicators (KPIs).

Palabras clave: DoS, Dataset, Aprendizaje automático

Firma del alumno:	VºBº Tutor/es:

Fernando Fernández Iglesias

Índice de contenidos

Índice de figuras

Índice de tablas

Glosario de términos

Dataset: Conjunto de datos estructurados que se utilizan para entrenar, validar o probar modelos de machine learning u otras técnicas de análisis de datos.

Kpi (Key Performance Indicator): Indicador clave de rendimiento. Un KPI es una métrica que se utiliza para medir el rendimiento o el éxito de un proceso o actividad en una organización.

DoS (Denial of Service): Ataque informático que tiene como objetivo provocar la indisponibilidad de un servicio o recurso de una red o sistema, sobrecargándolo con solicitudes maliciosas.

Machine learning (Aprendizaje automático): Rama de la inteligencia artificial que se enfoca en el desarrollo de algoritmos y técnicas que permiten a las computadoras aprender patrones y tomar decisiones basadas en datos, sin ser programadas explícitamente.

Router Core: Un dispositivo de red de alto rendimiento que dirige el tráfico de datos entre diferentes redes o subredes en una infraestructura de red, especialmente diseñado para redes empresariales y de proveedores de servicios de Internet.

DOROTHEA: DOROTHEA es una solución basada en Docker que permite crear topologías de red virtuales para generar y recopilar datos de flujo. Utiliza un sensor de NetFlow que recopila los flujos generados a partir de los paquetes que pasan por una interfaz de red.

KNN (K-Nearest Neighbors): Algoritmo de machine learning utilizado principalmente para clasificación y regresión. Clasifica puntos de datos en función de la mayoría de sus "vecinos" más cercanos en el espacio de características.

RF (Random Forest): Bosque aleatorio. Un algoritmo de aprendizaje automático que crea múltiples árboles de decisión y combina sus resultados para mejorar la precisión y reducir el sobreajuste.

SVM (Support Vector Machine): Máquina de vectores de soporte. Algoritmo de aprendizaje supervisado utilizado para la clasificación y regresión que encuentra un hiperplano óptimo que separa los datos en distintas clases.

DT (Decision Tree): Árbol de decisión. Un modelo de representación visual en forma de árbol que toma decisiones a partir de múltiples condiciones y reglas.

LR (Logistic Regression): Regresión logística. Un algoritmo de aprendizaje supervisado utilizado para la clasificación binaria o multinomial basado en la función logística.

NB (Naive Bayes): Clasificador Naive Bayes. Un algoritmo de clasificación probabilístico basado en el teorema de Bayes, que asume independencia entre las características.

XGB (XGBoost): Extreme Gradient Boosting. Un algoritmo de aprendizaje automático basado en el método de impulso de árboles de decisión, diseñado para mejorar la eficiencia y la precisión de los modelos de machine learning.

DJ (Dijkstra): Algoritmo utilizado para encontrar el camino más corto en un grafo ponderado desde un nodo de origen a todos los demás nodos en el grafo. Es comúnmente utilizado en problemas de optimización de rutas y redes.

GB (Gradient Boosting): Impulso de gradiente. Una técnica de aprendizaje automático que construye múltiples modelos débiles (por ejemplo, árboles de decisión) de manera secuencial para mejorar el rendimiento general del modelo.

REP tree (Repeated Incremental Pruning to Produce Error Reduction (REP) Tree): Un algoritmo de aprendizaje automático que construye árboles de decisión utilizando un proceso iterativo de poda para reducir el error.

DS (Decision Stump): Un árbol de decisión con solo un nodo de decisión y dos hojas, utilizado comúnmente como componente base en algoritmos de conjunto como el Adaboost.

RT (Regression Tree): Árbol de regresión. Similar al árbol de decisión, pero se utiliza para tareas de regresión en lugar de clasificación.

J48: Un algoritmo de aprendizaje automático que construye árboles de decisión basados en el algoritmo C4.5, utilizado para clasificación y regresión.

Introducción

En el contexto de la seguridad de redes, la detección de ataques de denegación de servicio (DoS) es un desafío crítico que ha perdurado a lo largo del tiempo. Si bien las redes de área local con sistemas de detección de intrusiones convencionales han abordado esta problemática, este trabajo de fin de máster se enfoca en un ámbito aún más desafiante: la detección de ataques DoS en redes de área extensa mediante el análisis de datos de flujo generados por routers core de Internet.

En este trabajo, se exploran las posibilidades de identificar ataques de denegación de servicio a través del análisis de datos de flujo recopilados por herramientas como Dorothea. Además, se abordan las diferentes fases y enfoques llevados a cabo para alcanzar este objetivo. El estudio se enfoca en el uso de técnicas de aprendizaje automático como una herramienta poderosa para esta tarea.

Planteamiento del problema

El problema de la detección de ataques de denegación de servicio (DoS) es crítico en el contexto de la seguridad de las redes. A pesar de las soluciones existentes para contrarrestar los ataques DoS conocidos, la constante evolución de las tácticas de los atacantes plantea nuevos desafíos en la detección temprana y precisa de estos ataques. En la actualidad, las redes de área local con sistemas de detección de intrusiones convencionales han abordado esta problemática con un grado de éxito relativo.

Sin embargo, la creciente sofisticación de los ciberdelincuentes hace que los sistemas tradicionales puedan no ser suficientes para detectar ataques encubiertos o de día cero. La necesidad de una detección más avanzada y eficaz se vuelve evidente, especialmente en redes de área extensa como Internet, donde la magnitud y la diversidad de los flujos de datos son considerables.

En este contexto, el presente trabajo se centra en la utilización del aprendizaje automático (Machine Learning) como una herramienta prometedora para abordar el problema de la detección de DoS. Los algoritmos de Machine Learning permiten

analizar grandes volúmenes de datos en tiempo real, identificando patrones y anomalías sutiles asociadas a ataques en desarrollo. Esta investigación busca no solo complementar los sistemas de detección tradicionales con el Machine Learning, sino también explorar la detección de DoS en redes de área extensa mediante el análisis de datos de flujo, un desafío que plantea importantes implicaciones para la seguridad de Internet.

Objetivos

El objetivo principal de este proyecto es realizar un análisis de denegaciones de servicio en flujos de datos, utilizando aprendizaje automático para su detección. Los objetivos específicos asociados con este proyecto son:

- Analizar el estado actual del problema. Para ello, se llevará a cabo un estudio general y una revisión de literatura, buscando los algoritmos de aprendizaje automático más efectivos utilizados para la detección de denegaciones de servicio.
- Investigar herramientas disponibles para simular denegaciones de servicio.
- Estudiar e implementar la herramienta seleccionada en Dorothea.
- Recolectar de los datos generados a través de Dorothea.
- Preprocesamiento de los datos recolectados.
- Aplicar los algoritmos de aprendizaje automático sobre los datos.
- Analizar de los resultados obtenidos.

Metodología

El proyecto que estamos abordando se ha estructurado en distintas fases, cada una con metodologías adaptadas a sus necesidades específicas. A continuación, desglosaremos el enfoque y la estructura de cada fase:

- **Investigación y Revisión de Literatura:** Esta fase se centró en un análisis preliminar sobre las denegaciones de servicio (DoS), así como en la revisión de la literatura referente al campo del aprendizaje automático aplicado a la detección de Denegaciones de servicio. Para la realización de esta revisión, se utilizaron recomendaciones del mapeo sistemático propuesto por Kitchenham, Budgen y Brereton[1].
- **Desarrollo del Proyecto:** El núcleo del proyecto reside en esta fase, en la cual se describen las distintas etapas que se han llevado a cabo para realizar el estudio y los resultados obtenidos. Con el fin de asegurar flexibilidad y eficacia en el proyecto, adoptamos la metodología ágil SCRUM[2], una metodología que permite una gran adaptabilidad del proyecto y facilita la identificación y solución de las incidencias que pueden surgir durante el proceso de desarrollo.

Estructura del trabajo

Este apartado se detalla el contenido de cada una de las secciones del documento y su descripción.

- **Introducción:**
  Es la sección actual, se presenta brevemente la estructura principal del proyecto y se detallan los objetivos y motivos para la realización de este trabajo, describiéndose también la estructura del documento.
- **Capítulo 1. Estudio del problema:**
  En este capítulo se define el contexto del problema y se realiza la revisión de literatura.
- **Capítulo 2. Gestión del proyecto:**

En este capítulo se trata la planificación del proyecto para desarrollar la herramienta. El cual incluye el alcance del proyecto, la gestión de tareas y la gestión de riesgos

- **Capítulo 3. Solución:**
  En este capítulo se describen la solución y las fases llevadas a cabo para realizar el análisis.
- **Capítulo 4. Evaluación:**
  En este capítulo se detallan las herramientas utilizadas para comprobar la validez de la solución encontrada, así como diversas tablas y gráficos que muestran los resultados obtenidos, además de una evaluación de los resultados obtenidos.
- **Capítulo 5. Conclusión:**
  En esta sección, se detallan las ideas y conceptos que se derivan de los resultados expuestos en el trabajo, así como los problemas encontrados y la previsión del futuro.
- **Anexo A. Control de versiones:**
  En este anexo se detalla los controles de versiones usados y cómo se usaron.
- **Anexo B. Planificación del proyecto:**
  Incluye el diagrama de Gantt la planificación del proyecto.

Capítulo 1: Estudio del problema

En este primer capítulo, se establece el contexto del problema en el apartado 1.1, donde se aborda la problemática central de los ataques de Denegación de Servicio (DoS) y se introduce la aplicación de técnicas de aprendizaje automático en la detección de estos ataques a través del análisis del tráfico de red. A continuación, en el apartado 1.2, se profundiza en la definición del problema, destacando los desafíos clave en la detección de ataques DoS. El apartado 1.3 se dedica al análisis más detallado del problema, incluyendo actores, motivación, objetivos y métodos de ataque, mientras que el apartado 1.4, "El estado de la cuestión", describe el propósito principal del proyecto y la importancia de comprender las tendencias actuales en el campo de la detección de ataques de DoS, lo que conduce a una revisión exhaustiva de la literatura existente.

1.1 El contexto del problema

Este proyecto se enfoca en el problema de las Denegaciones de Servicio (DoS), que consisten en un tipo de ataque cibernético que busca interrumpir o degradar la disponibilidad de un servicio o recurso en línea. El objetivo del trabajo es estudiar las posibilidades de detección de ataques de DoS mediante la aplicación de técnicas de aprendizaje automático en el análisis del tráfico de red.

El aprendizaje automático es una rama de la inteligencia artificial que se utiliza para analizar grandes conjuntos de datos. Su objetivo es construir sistemas que mejoren con la experiencia y sean capaces de generalizar comportamientos a conjuntos de datos más grandes. Esta tecnología se basa en el uso de algoritmos y heurísticas para analizar datos y está estrechamente relacionada con el reconocimiento de patrones.

En el análisis de datos, se utilizan técnicas de aprendizaje automático para resolver problemas como la regresión, clasificación, agrupamiento (o clustering) y reducción de la dimensionalidad. Dependiendo del problema a resolver, se utilizan diferentes tipos de algoritmos, como el aprendizaje supervisado, no supervisado y semisupervisado.

Este proyecto se ha centrado en técnicas de aprendizaje supervisado y, específicamente, en la clasificación de datos. El problema de clasificación consiste en predecir las categorías o clases a las que pertenecen los datos. En el caso de los ataques de DoS, disponemos de conjuntos de datos que contienen características del tráfico de red. Los datos se etiquetan como "Tráfico no DoS" y "Tráfico DoS" para clasificar los conjuntos de datos disponibles. Utilizamos un conjunto de datos previamente etiquetado para entrenar modelos de aprendizaje automático y, a partir de estos modelos, buscamos detectar patrones en conjuntos de datos no etiquetados para clasificarlos de manera precisa.

El siguiente punto importante es comprender el concepto de Denegaciones de Servicio (DoS). Este tipo de ataques busca interrumpir o degradar la disponibilidad de un servicio o recurso en línea.

Los servidores web tienen un límite de capacidad para procesar solicitudes simultáneas. Si se supera ese límite, el servidor puede volverse lento, dejar de responder o bloquearse.

Existen dos tipos de ataques de denegación de servicio:

- Ataque DoS (Denial of Service): Se generan numerosas solicitudes desde una única máquina o dirección IP, agotando los recursos del servicio objetivo.
- Ataque DDoS (Distributed Denial of Service): Se utilizan múltiples máquinas infectadas para llevar a cabo solicitudes simultáneas. Estas máquinas, que forman parte de una botnet controlada por un ciberdelincuente, han sido previamente comprometidas a través de la infección por malware como podemos observar en la figura 1.1.

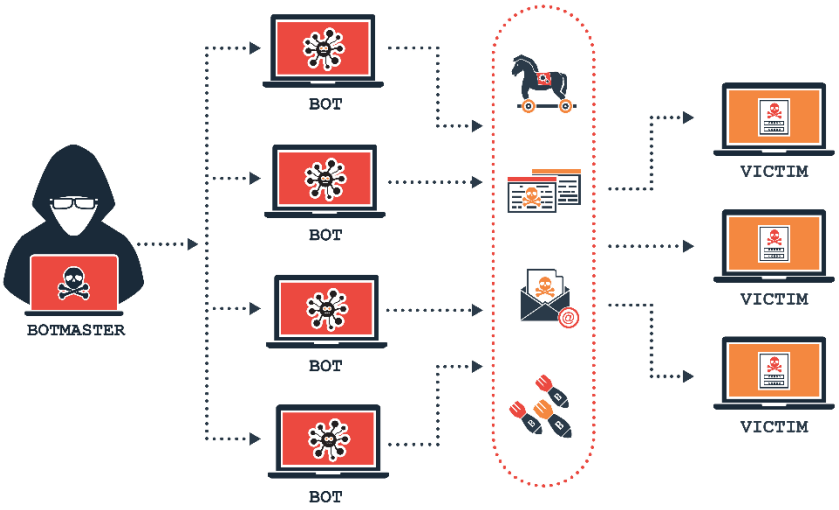

Figura 1.1. Infección de los ordenadores víctimas mediante malware

Fuente:https://www.incibe.es/ciudadania/blog/que-son-los-ataques-dos-y-ddos#:~:text=Existen%20dos%20t%C3%A9cnicas%20de%20este,s%20que%20realizan%20el%20ataque.

En este proyecto, buscamos detectar y prevenir ataques de Denegación de Servicio (DoS) mediante el análisis del tráfico de red. Esto implica identificar patrones anómalos o características distintivas en los conjuntos de datos que corresponden a ataques de DoS. Utilizamos técnicas de aprendizaje automático para entrenar modelos capaces de reconocer estos patrones y clasificar el tráfico de red en " Tráfico no DoS" o " Tráfico DoS ".

Este trabajo se ha llevado a cabo con el apoyo del Grupo de Robótica de la Universidad de León. Para la realización del proyecto se han empleado dos conjuntos de datos construidos a partir de una herramienta llamada Dorothea.

1.2 La definición del problema

El núcleo principal del problema radica en la detección de ataques DoS. Los principales desafíos a la hora de detectar estos ataques se detallan a continuación:

- **Volumen masivo de tráfico**: Los ataques DDoS suelen implicar una enorme cantidad de tráfico no deseado que puede sobrecargar rápidamente los servidores y redes, dificultando la identificación y bloqueo de las direcciones IP ofensivas.

- **Distinción entre tráfico legítimo y malicioso:** Uno de los principales desafíos de la detección de DDoS es diferenciar entre el tráfico legítimo y el tráfico malicioso. Los atacantes a menudo usan técnicas como IP spoofing (falsificación de IP) para hacer que el tráfico parezca legítimo.
- **Ataques distribuidos:** En un ataque DDoS, el tráfico malicioso proviene de múltiples fuentes, lo que puede hacer que sea extremadamente difícil rastrear la fuente original del ataque.
- **Ataques sofisticados y cambiantes:** Los ataques DDoS se han vuelto más sofisticados con el tiempo, con atacantes que cambian constantemente sus tácticas para evitar la detección. Por ejemplo, los ataques pueden incluir múltiples vectores de ataque, cambiar de dirección IP o fluctuar en volumen para eludir los controles de seguridad.
- **Recursos limitados:** En comparación con los atacantes, que pueden aprovechar redes de bots masivas para generar tráfico, las empresas a menudo tienen recursos limitados para detectar y mitigar los ataques.
- **Tiempo de respuesta:** La rapidez con la que una organización puede detectar y responder a un ataque DDoS puede ser crítica. Un retraso en la respuesta puede causar un tiempo de inactividad significativo y una interrupción del servicio.

1.3. Análisis del problema

En esta sección, se realizará un análisis más detallado de este tipo de ataques y de los problemas que presentan para el mundo de la ciberseguridad.

1.3.1. Actores, Motivación y Objetivos

En el mundo de la ciberseguridad, un ataque de DoS tiene tres aspectos clave a considerar: los responsables del ataque, las víctimas y los motivos detrás del ataque.

Respecto a los autores de los ataques de DoS, estos suelen ser:

- **Hacktivistas**: Son grupos o individuos movidos por causas políticas o sociales que recurren a los ataques de DoS para protestar o manifestarse.

- **Ciberdelincuentes**: Buscan ganancias económicas mediante ataques de DoS, utilizando tácticas como el chantaje o la extorsión.
- **Gobiernos**: Organizaciones estatales o patrocinadas por el estado pueden efectuar ataques de DoS con metas políticas o estratégicas.

Las razones detrás de los ataques de DoS suelen ser las siguientes:

- **Interrupción de servicio**: Algunos atacantes buscan interrumpir o disminuir la disponibilidad de un servicio en línea, ocasionando pérdidas económicas o dañando la reputación de una organización.
- **Activismo**: Los hacktivistas pueden usar los ataques de DoS como una forma de protestar contra una entidad que consideran reprochable.
- **Beneficio económico**: Los ciberdelincuentes pueden realizar ataques de DoS con la intención de extorsionar a las víctimas o como una estrategia de distracción para perpetrar otros tipos de ataques, como el robo de datos.
- **Guerra cibernética**: Los ataques de DoS pueden ser parte de operaciones cibernéticas estratégicas por parte de los gobiernos para debilitar a un enemigo.

Los objetivos habituales de los ataques de DoS suelen ser sistemas o servicios en línea con alto valor, ya sea económico u otro. Algunos de estos objetivos incluyen:

- **Websites populares**: Como los de organizaciones gubernamentales, instituciones financieras, empresas líderes en la industria, etc.
- **Infraestructura esencial**: Sistemas y servicios cruciales para el funcionamiento de un país, como las redes de energía, transporte, comunicaciones, etc.
- **Proveedores de servicios en línea**: Plataformas de e-commerce, redes sociales, servicios financieros online, etc., que son blancos de ataques de DoS debido a su relevancia y el impacto que pueden generar en los usuarios.
- **Rivales comerciales:** Algunas organizaciones pueden recurrir a ataques de DoS con el propósito de dañar a sus competidores y obtener ventajas comerciales.

1.3.2. Métodos y técnicas de ataque

Los ataques de denegación de servicio (DoS) se pueden clasificar en tres categorías principales: ataques a las capas de aplicación, ataques de protocolo y ataques volumétricos[3].

- **Ataques a las capas de aplicación**: Estos ataques están diseñados para explotar las vulnerabilidades específicas de un sistema, como los servicios de voz SIP, servidores web y BGP. Los ataques de capa de aplicación requieren menos recursos, ya que imitan el comportamiento legítimo de los usuarios, lo que dificulta su identificación. Los ataques dirigidos a servidores DNS y las inundaciones cifradas HTTP/S son subcategorías populares de este tipo de ataque.
- **Ataques de protocolo**: Los ataques de protocolo agotan los recursos del servidor y del equipo de comunicación que funciona como intermediario. Los piratas informáticos sobrecargan los recursos del servidor con solicitudes de protocolos falsos. Dentro de esta categoría, encontramos el ping de la muerte, las inundaciones SYN, la inundación Tsunami SYN y el agotamiento de conexión.
- **Ataques volumétricos**: Los ataques volumétricos agotan el ancho de banda de un sitio web usando métodos de amplificación. Se caracterizan por su naturaleza sigilosa, ya que se registran como tráfico auténtico generado por múltiples direcciones IP. Los ataques volumétricos más notorios incluyen la amplificación del DNS, la inundación UDP, la inundación ICMP o Ping y la inundación RST-FIN.

1.3.3. Ejemplos reales

Esta sección detalla tres de los ataques DoS más impactantes de los últimos años, brindando una visión más profunda sobre la naturaleza de estas amenazas.

1. AWS, febrero de 2020[4]**:**

Amazon Web Services (AWS) neutralizó en febrero de 2020 un importante ataque DDoS que generó un tráfico pico de 2.3 Tbps. El ataque, dirigido a un cliente no revelado por AWS, se realizó a través de servidores web del protocolo CLDAP, comprometidos para el efecto.

2. GitHub, febrero de 2018[5]**:**

GitHub, plataforma de gestión de código muy utilizada, fue objetivo de un masivo ataque DDoS en febrero de 2018. El ataque llegó a 1.3 Tbps y 126.9 millones de paquetes por segundo. Los atacantes emplearon el sistema de almacenamiento en caché memcached para amplificar su ataque, multiplicando su tamaño original hasta casi 50,000 veces. Gracias al servicio de protección contra DDoS de GitHub, el ataque fue mitigado en tan solo 10 minutos.

3. Dyn, 2016[6]**:**

Dyn, esencial proveedor de DNS, fue víctima en octubre de 2016 de un ataque DDoS que interrumpió el servicio de sitios web de gran importancia, como Airbnb, Netflix, PayPal, entre otros. El ataque se orquestó con el malware Mirai, que forma una botnet con dispositivos IoT vulnerables y los programa para enviar solicitudes a una única víctima. A pesar de la magnitud del ataque, Dyn logró mitigarlo en un día. Las motivaciones detrás del ataque, aunque no esclarecidas, podrían estar vinculadas a hacktivistas.

1.4. El estado de la cuestión

El objetivo principal de este proyecto es desarrollar una herramienta que permita la implementación de algoritmos de aprendizaje automático en conjuntos de datos generados por el tráfico de red. El propósito es detectar y prevenir ataques de denegación de servicio (DoS). Para conseguirlo, es crucial entender las tendencias actuales en esta área tecnológica. En este contexto, se ha llevado a cabo una revisión exhaustiva de la literatura existente relacionada con el aprendizaje automático y la detección de anomalías vinculadas a los ataques de DoS. Los hallazgos de esta revisión han sido la base para el desarrollo de la herramienta propuesta en este proyecto, así como para la ejecución de los experimentos necesarios.

La revisión de literatura tiene como objetivo responder a la siguiente pregunta de investigación:

RQ1: ¿Cuáles son las tendencias actuales en la aplicación de algoritmos de aprendizaje automático y conjuntos de datos de tráfico de red para la detección de tráfico anómalo generado por ataques de DoS?

Para llevar a cabo este estudio, se han seguido las pautas sugeridas por Kitchenham[1] y se ha empleado la guía PRISMA[7]. Además, se ha seguido una metodología basada en las recomendaciones de Kitchenham, Budgen y Brereton. Dicha metodología se divide en tres fases:

- Planificación de la búsqueda.
- Proceso de búsqueda y selección de estudios.
- Extracción de datos y elaboración del informe.

En los siguientes apartados se proporcionará información más detallada sobre la revisión de literatura realizada, desde la metodología empleada hasta el análisis de los resultados obtenidos.

1.4.1. Planificación de la búsqueda

En primer lugar, es importante mencionar que nuestro estudio ha implicado la extracción de variables tanto cualitativas como cuantitativas, lo que significa que es un estudio de tipo mixto.

El objetivo ha sido llevar a cabo un estado del arte por medio de una revisión sistemática en el tema de la detección de ataques de denegación de servicio que emplean técnicas y procedimientos para generar tráfico anómalo en las redes. Para lograrlo, hemos decidido considerar aquellos artículos que tratan específicamente sobre la detección de este tipo de tráfico.

El resultado esperado de la búsqueda de muestras de estudio es obtener entre veinte y treinta artículos. Una vez obtenido este conjunto inicial, aplicaremos varias etapas de filtrado para reducir el número a un subconjunto de, aproximadamente, seis artículos.

Para buscar los artículos, hemos seleccionado varias fuentes de información, como se indica a continuación:

- IEEE Xplore
- Scopus

1.4.2. Proceso de búsqueda

Una vez seleccionadas las fuentes de información para las búsquedas, es esencial elaborar cadenas de búsqueda efectivas que proporcionen resultados satisfactorios. Para esta tarea, se diseña una cadena de búsqueda que se aplicará en cada una de las fuentes de información seleccionadas. Estas cadenas se basan en palabras clave obtenidas a través de la aplicación de la estrategia PICOC [8] [9]a la pregunta de investigación.

Aunque la estrategia PICOC proviene del campo de las Ciencias de la Salud, se ha adaptado al ámbito de la Informática, omitiendo los campos "Comparación" y "Contexto". Por lo tanto, se utiliza la siguiente adaptación:

- **Población**: algoritmos; conjuntos de datos
- **Intervención**: detección de DoS; aprendizaje automático; sistema de detección de intrusiones
- **Resultado**: métricas; características

Después de obtener los resultados de esta estrategia, se traducen al inglés y se buscan sinónimos relevantes para formar la siguiente cadena de búsqueda:

- ("denial of service attack" OR "DoS attack" OR "denial-of-service attack" OR "DoS detection") AND ("machine learning algorithms" OR "classification" OR "intrusion detection system" OR "IDS") AND ("datasets")

Utilizando estas cadenas de búsqueda, se intenta recopilar un conjunto de artículos relevantes para la investigación. Durante el proceso de búsqueda, se aplica un filtro relacionado con la fecha de publicación de los artículos. Dado que el campo de la ciberseguridad evoluciona rápidamente, solo se consideran los artículos publicados entre 2020 y 2023. Además, se centra en artículos de acceso libre.

1.4.3. Proceso de selección de las muestras

Para gestionar la selección de las muestras recogidas, se instauró un proceso de cribado. En primer lugar, se descartaron los artículos repetidos. Posteriormente, se llevó a cabo un filtrado basado en la lectura de los resúmenes de los trabajos restantes. Para decidir si se acepta o rechaza un artículo específico, se fijaron varios criterios de inclusión y exclusión. Si un artículo satisface un criterio de exclusión y uno de inclusión, se otorga mayor relevancia al criterio de exclusión. Por consiguiente, se prescinden de aquellos artículos que cumplan al menos un criterio de exclusión o que no cumplan ninguno de los criterios de inclusión.

Los criterios de inclusión son los siguientes:

- **CI1**: El artículo trabaja con conjuntos de datos de tráfico de red.
- **CI2**: El artículo trata sobre las denegaciones de servicio (DoS), ya sea enfocándose en su estudio o mencionándolas.
- **CI3**: El artículo utiliza varias técnicas de aprendizaje automático aplicadas al análisis de tráfico de red.
- **CI4**: El artículo hace referencia a los algoritmos utilizados en el aprendizaje automático.

Los criterios de exclusión son los siguientes:

- **CE1**: El artículo no utiliza técnicas de aprendizaje automático.
- **CE2**: El artículo no pertenece al campo de la ciberseguridad.
- **CE3**: El artículo es una revisión de literatura.
- **CE4**: La publicación no ha sido citada al menos en una ocasión.
- **CE5** El artículo emplea técnicas de aprendizaje automático aplicadas a la ciberseguridad, pero no las aplica al análisis de tráfico de red.
- **CE6**: El artículo no ha sido publicado entre el 2020 y 2023.

Con la finalidad de examinar cerca de seis artículos, se ha elaborado una herramienta de valoración de la calidad del artículo, fundamentada en ciertas demandas de relevancia para este estudio. Este instrumento de evaluación se ha llenado con la información recopilada tras la lectura íntegra de los artículos escogidos en el paso previo. La herramienta facilita la valoración de los artículos antes de pasar a la recolección de resultados e incluye las siguientes cuestiones:

- **P1**: ¿Utiliza varios algoritmos de aprendizaje automático?
- **P2**: ¿Hace referencia al problema de las denegaciones de servicio (DoS)?
- **P3**: ¿Utiliza técnicas de aprendizaje automático?
- **P4**: ¿Especifica el o los conjuntos de datos con los que trabaja?
- **P5**: ¿Incluye gráficos y/o métricas para tratar los distintos tipos de variables?

Estas preguntas se han elegido en función de las características de los artículos que resultan más relevantes para la investigación. El propósito de estas preguntas es distinguir aquellos artículos centrados en la aplicación de algoritmos de aprendizaje automático en conjuntos de datos de tráfico de red. Tras delinear las preguntas, es esencial determinar las posibles respuestas a dichas interrogantes y el valor de cada respuesta. Para la elaboración de este cuestionario, se ha buscado plantear todas las cuestiones de tal manera que admitan una respuesta de Sí o No, asignando a dichas respuestas 1 punto o 0 puntos, respectivamente. De esta forma, la calificación máxima que un artículo puede obtener es de cinco puntos. Por último, para asegurar la calidad de los artículos escogidos, se ha fijado que el puntaje mínimo para que un artículo sea considerado es de 4 puntos.

1.4.4. Proceso de extracción de los datos

El último paso en la metodología propuesta implica determinar la estrategia de recolección de los datos relevantes para la investigación. Tras delimitar el ámbito de estudio a los artículos pertinentes para la investigación, se ha diseñado un formulario para la recolección de datos. Las variables a recolectar de cada artículo se derivan de la pregunta de investigación planteada al comienzo de esta revisión. Los datos se recolectarán a través de la lectura completa y minuciosa de cada uno de los artículos. Las variables deseadas se han establecido en línea con el objetivo de llevar a cabo una revisión sistemática, asegurando así la posibilidad de que este procedimiento de recolección sea replicable y objetivo. En este sentido, las variables que se recolectarán son las siguientes:

- **V1** Precisión del algoritmo o algoritmos empleados.
- **V2** Fecha del dataset o datasets utilizados.
- **V3** Numero de características del dataset (en caso de indicarse)
- **V4** Enumeración de los algoritmos aplicados en el estudio.
- **V5** Software de aprendizaje automático utilizado.

Los resultados obtenidos tras aplicar el formulario de recolección a los artículos seleccionados se discutirán en las secciones siguientes.

1.4.5. Resultados del proceso de selección

El proceso de selección de artículos para mi estudio comenzó con la recopilación de cerca de 60 trabajos que podrían ser relevantes. Las búsquedas que realicé en las bases de datos mencionadas en la sección 1.4.2 resultaron en un total de 140 artículos. De estos, seleccioné los más relevantes de cada base de datos. La cantidad de artículos que recopilé de cada fuente se puede ver en la tabla 1.1.

Fuente de información	Número de artículos
Scopus	33 resultados
IEEE Xplore	27 resultados

Tabla 1.1. Número de artículos obtenidos

Después de esta recopilación inicial, me quedé con 60 artículos. Pero tras detectar 2 duplicados, el total se redujo a 58. Seguidamente, apliqué los criterios especificados en la sección 1.4.3, identificados como CI para los criterios de inclusión y CE para los criterios de exclusión.

Estos criterios me permitieron reducir la selección a 16 artículos. Es decir, de los 58 artículos iniciales, descarté 42.

Finalmente, apliqué el cuestionario descrito en la sección 1.4.4. Esto resultó en descartar 10 de los 16 artículos, quedándome con los seis que obtuvieron las mejores puntuaciones. Así, el conjunto final de artículos que se examinarán en mi estudio es seis. Las referencias de estos artículos se pueden consultar en la tabla 1.2.

ID	Referencia	Año	Título
A1	[10]	2021	A machine learning approach for improving the performance of network intrusion detection systems
A2	[11]	2020	Evaluation of Classification algorithms for Distributed Denial of Service Attack Detection
A3	[12]	2022	A Machine Learning-Based Classification and Prediction Technique for DDoS Attacks
A4	[13]	2023	Man-in-the-middle and denial of service attacks detection using machine learning algorithms
A5	[14]	2022	Fast and accurate classifying model for denial-of-service attacks by using machine learning
A6	[15]	2020	Detection DDOS attacks using machine learning methods

Tabla 1.2. Identificador y referencia de los artículos seleccionados

1.4.6. Resultados del proceso de extracción de datos

En esta sección, presentaremos los hallazgos derivados del proceso de extracción de características, tal como se detalla en la subsección 1.4.4. Nuestro objetivo es responder a la pregunta de investigación RQ1, que se centra en identificar las tendencias actuales en la aplicación de algoritmos de aprendizaje automático y conjuntos de datos de tráfico de red para la detección de tráfico anómalo generado por ataques de denegación de servicio (DoS).

Los resultados obtenidos de los seis artículos seleccionados revelan información importante sobre los conjuntos de datos utilizados y los algoritmos implementados. En cuanto a los conjuntos de datos (Tablas 1.3.), los más citados son CICDoS2019 y CIC-IDS2017. Además, se hace referencia a un cuarto conjunto de datos recopilado de Kaggle en el artículo A4. Estos conjuntos de datos son relevantes y proporcionan una base sólida para llevar a cabo investigaciones en la detección de ataques de DoS.

Id	Fecha	N.º de características	Ref
CICDoS2019	2019	88	[11] [15]
CIC-IDS2017	2017	84	[10] [14]
UNSW-nb15	2015	49	[12]

Tabla 1.3. Datasets utilizados en los artículos

La tabla 1.4 muestra los diversos algoritmos de aprendizaje automático utilizados en los artículos seleccionados, así como la precisión obtenida con cada uno de ellos en la detección de ataques de DoS. Los algoritmos más utilizados como se puede ver en la Tabla 1.5. son Random Forest (RF), Support Vector Machines (SVM) y Decision Trees (DT). Otros algoritmos utilizados incluyen Logistic Regression (LR), Naive Bayes (NB), K-Nearest Neighbors (KNN), Gradient Boosting (GB), REP Tree, Decision Stump (DJ), Extreme Gradient Boosting (XGBoost), Random Tree (RT) y J48.

Articulo	**Algoritmo**	**Precisión**
A1	SVM, RF, DJ	98,18 %, 96,76 %, 96,50 %
A2	DT, NB, LR, SVM, KNN, RF	99.99 %, 97.72 %, 79.34 %, 50.1 %, 100 %, 100 %
A3	RF, XGB	89 %, 90 %
A4	XGB, RF, DT, GB	97,8 %, 97.2 %, 97.2 %, 97.6 %
A5	REP tree, DS, RT, RF, J48	99,95 %, 81,57 %, 99,96 %, 99,9 %, 99,9 %
A6	LR, NB, KNN, DT, RF, SVM	99,8 %, 98,7 %, 99,9 %, 99 %, 98,4 %, 99,7 %

Tabla 1.4. Algoritmos y precisión de los artículos

En términos de precisión, varios de los algoritmos lograron un rendimiento superior al 99% en al menos un estudio. Estos algoritmos incluyen RF, SVM, KNN y REP Tree. Sin embargo, es importante destacar que la precisión puede variar

según el conjunto de datos utilizado y cómo se preprocesen y seleccionen las características de los datos.

Algoritmo	**Porcentaje de aparición**
RF	83.33%
SVM	50.00%
DT	50.00%
LR	33.33%
NB	33.33%
KNN	33.33%
XGB	33.33%
DJ	16.67%
GB	16.67%
REP tree	16.67%
DS	16.67%
RT	16.67%
J48	16.67%

Tabla 1.5. Porcentaje de aparición de los algoritmos

En cuanto al software utilizado como se puede observar en la Tabla 1.6, los artículos seleccionados utilizaron diversas herramientas y bibliotecas. Algunos de los softwares mencionados incluyen Azure Machine Learning Tool, Scikit-learn, XGBoost y Weka, entre otros. Estas herramientas proporcionan funcionalidades y algoritmos necesarios para implementar y evaluar los modelos de detección de tráfico anómalo.

Dentro de los artículos el se puede observar que el más utilizado es Scikit-learn, la cual es una biblioteca de aprendizaje automático de código abierto que ofrece una amplia gama de algoritmos y herramientas para el análisis de datos y la construcción de modelos predictivos. Su popularidad se debe a su facilidad de uso, eficiencia computacional y a la diversidad de algoritmos que proporciona.

Articulo	Software
A1	Azure Machine Learning Tool
A2	Scikit-learn
A3	Scikit-learn
A4	Scikit-learn, XGBoost
A5	Scikit-learn
A6	Scikit-learn

Tabla 1.6. Software utilizado

Tras completar esta revisión sistemática, podemos observar que los resultados obtenidos indican que los conjuntos de datos más utilizados para la clasificación del tráfico de red son CICDoS2019 y CIC-IDS2017. Por tanto, estos conjuntos de datos pueden ser de gran utilidad para desarrollar sistemas de detección de tráfico anómalo provocado por ataques de denegación de servicio (DoS).

A partir de la extracción de datos, se ha concluido que los algoritmos de aprendizaje automático que ofrecen mayor precisión son Random Forest (RF), Support Vector Machines (SVM) y Decision Trees (DT), aunque otros algoritmos como Logistic Regression (LR), Naive Bayes (NB), K-Nearest Neighbors (KNN), entre otros, también han demostrado ser eficaces.

Es importante señalar que, al desarrollar una investigación en el ámbito del aprendizaje automático, la elección de la herramienta es crucial. En este estudio, se ha observado que la tendencia actual se inclina hacia el uso de Scikit-learn. Esta biblioteca de aprendizaje automático de código abierto ofrece una amplia gama de algoritmos y herramientas para el análisis de datos y la construcción de modelos predictivos, y su popularidad se debe a su facilidad de uso, eficiencia computacional y a la diversidad de algoritmos que proporciona.

No obstante, estos resultados, a pesar de mostrar altos niveles de precisión, no constituyen una solución definitiva al problema de la detección de tráfico anómalo. sino que se intenta proporcionar un contexto que pueda resultar de utilidad a la hora de alcanzar el objetivo planteado en el presente proyecto.

Además, no se han estudiado en detalle fenómenos como el sobreajuste y el subajuste, que son aspectos fundamentales que considerar en la aplicación de estos algoritmos y merecen un análisis más exhaustivo.

Capítulo 2: Gestión del proyecto

En este segundo capítulo, se aborda la gestión integral del proyecto. En el apartado 2.1, se define el alcance del proyecto y se establecen los objetivos finales junto con los pasos necesarios para su consecución. A continuación, en el apartado 2.2, se presenta una planificación detallada que abarca desde la investigación hasta la elaboración del informe técnico, siguiendo la metodología ágil Scrum con seis Sprint que comprenden tareas específicas para avanzar continuamente en el proyecto. El apartado 2.3 se enfoca en la gestión de recursos, especificando los recursos humanos, materiales y económicos necesarios. En el apartado 2.4, se abordan los riesgos identificando sus categorías y proporcionando análisis de probabilidad, impacto, causas y soluciones, con el objetivo de establecer estrategias de mitigación. Por último, en el apartado 2.5, se hace mención de la legislación y normativa aplicable, asegurando el cumplimiento de los requisitos legales durante el desarrollo del estudio.

2.1. Alcance del proyecto

2.1.1. Definición del proyecto

El objetivo final de este proyecto es realizar un análisis de datos de flujo, específicamente de denegaciones de servicio. Para ello, se implementó una denegación de servicio en la herramienta llamada Dorothea. Dorothea permite simular tanto tráfico de red benigno como malicioso, capturando así el conjunto de datos necesario para el estudio.

Una vez capturados varios conjuntos de datos (datasets), se procederá a su tratamiento utilizando técnicas de preprocesamiento. Posteriormente, se dividirán en datos de entrenamiento y prueba para aplicar distintas técnicas de aprendizaje automático. De esta forma, se podrán observar los indicadores clave de rendimiento y realizar un análisis de los mismos.

2.1.2. Estimación de tareas y recursos

2.1.2.1 Tareas

Este proyecto sigue la metodología ágil Scrum, en la que se realizarán 6 sprints que incluirán las tareas detalladas en la planificación del proyecto.

2.1.2.2 Recursos

Los recursos necesarios para este análisis que se está realizando en este documento se dividen en las siguientes categorías:

- Recursos físicos: Esta categoría incluye una oficina para llevar a cabo las reuniones y encuentros necesarios, los costos de electricidad, y un ordenador personal con las siguientes especificaciones: procesador Intel Core i7-1255U, 8 GB de memoria RAM, 1 TB de almacenamiento SSD. En cuanto al software, no se requerirá ningún costo adicional, ya que todos los programas utilizados son de código abierto.
- Recursos Humano: Se pretende contratar a un equipo para el desarrollo del proyecto. Este equipo está formado por un Scrum Master, que guía al Scrum Team en el uso de la metodología; y un Scrum Team formado por un analista de seguridad informática y un analista de Datos.

2.1.3. Presupuesto

El presupuesto tiene como objetivo estimar los gastos desde el inicio hasta la finalización del proyecto. A continuación, se presenta el importe aproximado de cada uno de los recursos necesarios para llevar a cabo el proyecto.

Para calcular el presupuesto de recursos humanos, se considera el salario de cada miembro del equipo del proyecto, como se detalla en la Tabla 2.1. A continuación, se especifica la cantidad asociada a cada miembro del equipo:

- Scrum Master: Según la plataforma "Indeed", el salario medio para este puesto en España es de 18,22 € por hora[16].
- Analista de seguridad informática: De acuerdo con "Indeed", el salario medio en España para este rol es de 16,87 € por hora [17].

- Analista de Datos: Basándonos en la información proporcionada por "Indeed", el salario medio en España para este cargo es de 13,84 € por hora [18].

Rol	**Cantidad (horas)**	**Salario (€/hora)**	**Total (€)**
Scrum Master	18	18,22 €	327,96 €
Analista de seguridad informática	150	16.87 €	4.930,5 €
Analista de datos	75	13.84 €	1.038 €
		Total	6.296,46 €

Tabla 2.1. Presupuesto en recursos humanos

El presupuesto designado para gastos generales contempla los costes indirectos del proyecto, esenciales para su desarrollo. Estos abarcan elementos como el alquiler del local, los servicios de luz e internet, el hardware necesario, material de oficina y servicios de limpieza. Estos gastos representarán un 15% del costo total del proyecto. Por otro lado, el beneficio industrial refleja las ganancias proyectadas, y para este caso, se estima que será del 11%.

El presupuesto total del proyecto se muestra en la Tabla 2.2.

Concepto	**Coste (€)**
Costes directos (Personal)	6.296,46 €
Costes indirectos (15%)	1.277,99 €
Beneficio Industrial (11%)	937,19 €
Subtotal	**8.519,92 €**
Iva aplicable (21%)	1.779,78 €
Total	**10.299,70 €**

Tabla 2.2. Presupuesto total

2.2. Plan de trabajo

En este apartado se identificarán y definirán las tareas e hitos del desarrollo del proyecto.

2.2.1. Identificación de las tareas

- Investigación
  - Investigación sobre el estado actual de las denegaciones de servicio (DoS).
  - Búsqueda detallada de herramientas para ataques DoS.
  - Estudio sobre la aplicación de aprendizaje automático en la detección de denegaciones de servicio.
  - Identificación de los mejores algoritmos y enfoques actuales.
- Realización de pruebas de concepto
  - Configuración de un contenedor Docker con dos máquinas Ubuntu.
  - Ejecución de pruebas con las herramientas identificadas para causar denegación de servicio.
  - Selección de la herramienta de ataque para la recopilación de datos de flujo.
- Estudio e Implementación de Dorothea
  - Investigación sobre la herramienta Dorothea.
  - Implementación de la herramienta de denegación de servicio.
  - Creación de scripts en Python para iniciar y finalizar el ataque.
  - Pruebas para validar el ataque en Dorothea.
- Recopilación de Datos Benignos y de Denegación
  - Ejecución y recolección de datos con un atacante.
  - Ejecución y recolección de datos con cinco atacantes.
  - Ejecución y recolección de datos con diez atacantes.
  - Ejecución y recolección de datos con muestreo (sampling).
  - Recopilación de datos de flujo benignos.
- Tratamiento de los conjuntos de datos
  - Investigación Teórica del Aprendizaje Automático
  - Preprocesamiento de los datos recolectados.
  - Aplicación de algoritmos de aprendizaje automático.
  - Observación de indicadores de rendimiento.
- Elaboración de informe técnico

2.2.2. Descripción de las tareas

La metodología Scrum aplicada en este proyecto establece que cada sprint se compone de tareas independientes que contribuyen a la evolución y desarrollo continuo del proyecto. Cada fase de planificación conforma lo que se denomina un sprint en esta metodología. Cada sprint consta de una reunión de planificación, el propio sprint y una reunión de revisión.

A continuación, se describen las fases que se han realizado en este proyecto, explicando las tareas llevadas a cabo en cada una.

Investigación (Sprint 1):

Durante el primer sprint del proyecto, se llevó a cabo una investigación exhaustiva sobre el estado actual de las denegaciones de servicio (DoS) y se realizó una búsqueda detallada de diferentes herramientas utilizadas para llevar a cabo este tipo de ataques.

Además, se realizó un estudio sobre la aplicación de aprendizaje automático en la detección de denegaciones de servicio, con el objetivo de identificar los mejores algoritmos y enfoques utilizados en la actualidad.

Realización de pruebas de concepto (Sprint 2):

Después de llevar a cabo un estudio detallado sobre las herramientas utilizadas para realizar una denegación de servicio, en esta fase procedimos a configurar un contenedor Docker. Este contenedor albergaba dos máquinas con Ubuntu: una de ellas estaba designada para ejecutar el ataque, mientras que la otra actuaba como la máquina víctima.

Posteriormente, realizamos diversas pruebas de concepto utilizando las herramientas identificadas en la fase de investigación. El objetivo era comprobar que el servidor Apache de la máquina víctima dejara de responder o funcionar adecuadamente. Por último, se selecciona la herramienta de ataque que se utilizaría para la recopilación de datos de flujo.

Estudio e Implementación de Dorothea (Sprint 3):

Durante esta fase, se investigó el funcionamiento de la herramienta Dorothea, de la cual se obtuvieron conjuntos de datos benignos y de denegación

de servicio. Luego de este estudio, se implementó la herramienta de denegación de servicio correspondiente. Para ello, se añadió el código necesario y se crearon los scripts en Python encargados de iniciar y finalizar el ataque. Se llevaron a cabo diversas pruebas para verificar el correcto funcionamiento del ataque en Dorothea.

Recopilación de Datos Benignos y de Denegación (Sprint 4):

En esta fase, se llevarán a cabo múltiples ejecuciones de Dorothea con el objetivo de recopilar datos para su análisis. Las pruebas a realizar incluirán la recolección de datos provenientes de un atacante contra una víctima, cinco atacantes contra una víctima y diez atacantes contra una misma víctima. Se efectuarán ataques sin muestreo (sampling) y con un muestreo (sampling) definido en 1000. En este contexto, el muestreo se refiere a la recolección de un dato por cada mil registrados. Además, se recopilaron datos de flujo benignos para su uso futuro.

Tratamiento de los conjuntos de datos (Sprint 5):

En esta fase, después de realizar un estudio sobre cómo se realiza el aprendizaje automático, se procedió al tratamiento de los conjuntos de datos recopilados en el sprint anterior. Se llevó a cabo un preprocesamiento de estos datos, lo cual implicó la reducción de características y la eliminación de aquellas que no fueran relevantes. Además, se combinaron los datos benignos con los de denegación, etiquetando adecuadamente cada conjunto. Asimismo, se realizó la normalización de los datos y se dividió el conjunto total en datos de entrenamiento y datos de prueba. Por último, se aplicaron los algoritmos correspondientes de aprendizaje automático y se observaron los indicadores de rendimiento.

Elaboración de informe técnico (Sprint 6):

En la última fase del proyecto, nos centramos en la redacción y estructuración del informe técnico. Este informe compila y organiza toda la información recolectada y las conclusiones derivadas de las fases anteriores.

2.2.3. Planificación de las tareas.

En este punto se detalla el periodo de tiempo establecido para cada fase, conocida como "sprint" en la metodología ágil Scrum. Además, se mencionan los tiempos de desarrollo de cada tarea. Cada sprint incluye una reunión de planificación y una reunión de revisión.

A continuación, se presenta una especificación de la organización inicial del proyecto.

- **Investigación** (Sprint 1): 23/12/2022 – 02/02/2023
  Participantes: Scrum Master y Analista de seguridad informática
  - Reunión de planificación: 22/12/2022. Durante esta reunión se deciden las tareas que se van a realizar en el sprint. En este caso se inician las tareas de investigación.
  - Sprint: Las tareas en concreto que forman este sprint son la investigación sobre el estado actual de las denegaciones de servicio (DoS), búsqueda detallada de herramientas para ataques DoS, estudio sobre la aplicación de aprendizaje automático en la detección de denegaciones de servicio y la Identificación de los mejores algoritmos y enfoques actuales.
  - Reunión de revisión: 03/02/2023. En esta reunión se comprueba que todas las tareas se completaron de forma satisfactoria y se corrigen aquellas tareas que lo requieren.
- **Realización de pruebas de concepto** (Sprint 2): 05/02/2023 – 24/02/2023
  - Reunión de planificación: 04/02/2023. Durante esta reunión se deciden las tareas que se van a realizar en el sprint. En este caso se inician las tareas relacionadas con la realización de pruebas de concepto.
  - Sprint: Las tareas en concreto que forman este sprint son la configuración de un contenedor Docker con dos máquinas Ubuntu, ejecución de pruebas con las herramientas identificadas para causar denegación de servicio y selección de la herramienta de ataque para la recopilación de datos de flujo.

  - Reunión de revisión: 27/02/2023. En esta reunión se comprueba que todas las tareas se completaron de forma satisfactoria y se corrigen aquellas tareas que lo requieren.
- **Estudio e Implementación de Dorothea** (Sprint 3): 01/03/2023 – 21/03/2023
  - Reunión de planificación: 28/02/2023. Durante esta reunión se deciden las tareas que se van a realizar en el sprint. En este caso se inician las tareas de estudio e implementación de la herramienta de denegación de servicio en Dorothea.
  - Sprint: Las tareas en concreto que forman este sprint son la investigación sobre la herramienta Dorothea, implementación de la herramienta de denegación de servicio, creación de scripts en Python para iniciar y finalizar el ataque, además de pruebas para validar el ataque en Dorothea.
  - Reunión de revisión: 22/03/2023. En esta reunión se comprueba que todas las tareas se completaron de forma satisfactoria y se corrigen aquellas tareas que lo requieren.
- **Recopilación de Datos Benignos y de Denegación** (Sprint 4): 24/03/2023 – 13/04/2023
  - Reunión de planificación: 23/03/23. Durante esta reunión se deciden las tareas que se van a realizar en el sprint. En este caso se inician las tareas de recopilación de los datos de flujo.
  - Sprint: Las tareas en concreto que forman este sprint son la ejecución y recolección de datos con un atacante, ejecución y recolección de datos con cinco atacantes, ejecución y recolección de datos con diez atacantes, ejecución y recolección de datos con muestreo (sampling) y Recopilación de datos de flujo benignos
  - Reunión de revisión: 14/04/2023. En esta reunión se comprueba que todas las tareas se completaron de forma satisfactoria y se corrigen aquellas tareas que lo requieren.

- **Tratamiento de los conjuntos de datos** (Sprint 5): 18/04/2023 – 19/05/2023
    - Reunión de planificación: 17/04/2023. Durante esta reunión se deciden las tareas que se van a realizar en el sprint. En este caso se inician las tareas del tratamiento de los conjuntos de datos obtenidos.
    - Sprint: Las tareas en concreto que forman este sprint son Investigación Teórica del Aprendizaje Automático, Preprocesamiento de los datos recolectados, aplicación de algoritmos de aprendizaje automático, observación de indicadores de rendimiento.
    - Reunión de revisión: 22/05/23. En esta reunión se comprueba que todas las tareas se completaron de forma satisfactoria y se corrigen aquellas tareas que lo requieren.
- **Elaboración de informe técnico** (Sprint 6): 24/05/2023 -24/07/23
    - Reunión de planificación: 23/05/23. Durante esta reunión se deciden las tareas que se van a realizar en el sprint. En este caso se inician las tareas de elaboración del informe técnico.
    - Sprint: en este sprint se desarrolla el documento que contiene todos los datos y la información recopilada durante el estudio
    - Reunión de revisión: 25/07/2023. En esta reunión se comprueba que todas las tareas se completaron de forma satisfactoria y se corrigen aquellas tareas que lo requieren.

Una vez organizadas todas las tareas se presentan en el diagrama de Gantt el cual podemos encontrar en el Anexo B.

2.3 Gestión de recursos

En este apartado se asignarán y listarán los recursos necesarios para el desarrollo del proyecto.

2.3.1. Especificación de los recursos.

Humanos:

- Scrum Master
- Scrum Team:
    - 1 analista de seguridad informática.
    - 1 analista de datos.

Materiales:

- Oficina
    - Alquiler de las oficinas
    - Luz
    - Internet
    - Servicio de limpieza
    - Material de oficina
- 1 ordenador para cada integrante del equipo

Económicos:

- Un total de 10.299,70 €, los cuales se han especificado en el apartado del presupuesto

2.3.2. Asignación de los recursos

A cada miembro del equipo del proyecto se le asigna un ordenador personal, además de hacer uso de todo el material de la oficina, luz e internet.

2.4. Gestión de los riesgos

En este apartado se incluyen los riesgos y su mitigación con el fin de subsanarlos en caso de que sucedan.

2.4.1. Identificación de los riesgos

Se pueden clasificar en tres secciones: externos, técnicos o de la organización y dirección del proyecto:

- Riesgos externos
    - Problemas con el hardware de los ordenadores
    - Tecnologías de código abierto deja de serlo
- Riesgos técnicos
    - Limitación de recursos para gestionar y procesar datos extensos
    - Modificación o actualización importante de las librerías usadas
- Riesgos de la organización y dirección del proyecto
    - Incumplimiento de los objetivos
    - Desfases temporales respecto a la planificación
    - Perdida de datos o de información
    - Desequilibrio en la asignación presupuestaria para el proyecto.

2.4.2. Análisis de los riesgos

En la Tabla 2.3 podemos ver como se listan los riesgos detectados, así como la probabilidad, el impacto, la causa y la solución.

ID	Riesgo	Probabilidad	Impacto	Causa	Solución
R01	Problemas con el hardware de los ordenadores	Baja	Alto	Un ordenador deja de funcionar debido a un fallo en sus componentes, ocasionado por un defecto de fabricación.	Verificar si el ordenador está en garantía para poder arreglarlo y posponer las tareas correspondientes a la persona encargada de ese ordenador
R02	Tecnologías de código abierto de serlo	Bajo	Bajo	Una tecnología utilizada en el proyecto, que era de código abierto y por ende gratuita, ha dejado de serlo.	Verificar si existe alguna tecnología de código abierto similar a la que actualmente estamos utilizando, para considerar su posible reemplazo.
R03	Limitación de recursos para gestionar y procesar datos extensos	Media	Alto	Carencia de recursos para gestionar conjuntos de datos extensos y para crear, así como entrenar, modelos con una cantidad adecuada de datos.	Incrementar la capacidad de almacenamiento y procesamiento a través de la adquisición o alquiler de hardware más potente o considerar la utilización de servicios en la nube especializados en el manejo de grandes volúmenes de datos.
R04	Modificación o actualización importante de las librerías usadas	Baja	Alto	Alguna de las librerías usadas sufre alguna actualización que dejan obsoleto o modifican el comportamiento del código utilizado.	Comprobar la librería que ha sido actualizada y modificar el código necesario para que vuelva a funcionar con normalidad
R05	Incumplimiento de los objetivos	Media	Alto	Mala organización a la hora de llevar a cabo la tarea	Buscar una manera de organización para llevar a cabo los objetivos del proyecto.
R06	Desfases temporales respecto a la planificación	Media	Bajo	Falta de organización o problemas externos que afectaron a la organización	Aumentar el plazo para la realización de esa tarea.
R07	Perdida de datos o de información	Bajo	Alto	Pérdida de datos en la documentación o referentes al proyecto.	Intentar recuperar los documentos perdidos, si no se consigue volver a reescribirlos
R08	Desequilibrio en la asignación presupuestaria para el proyecto.	Media	Alto	Cambios en los requerimientos del proyecto que aumentan costos del presupuesto.	Elaboración de un presupuesto que permita realizar ajustes en la planificación y en la estimación de recursos.

Tabla 2.3 Análisis de riesgos

2.5. Legislación y normativa

En el marco de este proyecto de investigación en ciberseguridad enfocado en la detección de ataques de Denegación de Servicio (DoS) mediante el análisis de datos de flujo, es importante considerar las leyes y normativas relevantes que pueden aplicar en el contexto de la ciberseguridad:

Ley de Propiedad Intelectual (LPI)[19]**:** Aunque este proyecto se desarrolla en un entorno Docker aislado de internet, es fundamental cumplir con las leyes de propiedad intelectual si involucra el desarrollo de software, herramientas, o la creación de modelos de detección. Las leyes de propiedad intelectual pueden aplicar a la protección de los derechos de autor y propiedad intelectual relacionados con el software y los resultados de investigación.

Reglamento General de Protección de Datos (RGPD)[20]**:** A pesar de que este reglamento está principalmente enfocado en la protección de datos personales, es relevante en el contexto de la ciberseguridad. El RGPD establece requisitos para garantizar la seguridad de los datos personales, incluyendo medidas técnicas y organizativas. Aunque el proyecto no involucre datos personales, es importante mantener buenas prácticas de seguridad de datos.

Ley de Seguridad de las Redes y Sistemas de Información (Ley NIS)[21]**:** Aunque esta ley se centra en la seguridad de las redes y sistemas de información en infraestructuras críticas, es importante tener en cuenta su enfoque en la seguridad en línea. Si bien el proyecto se desarrolla en un entorno aislado, es fundamental evitar cualquier acción que pueda comprometer la seguridad de las redes y sistemas de información.

Capítulo 3: Solución

En este capítulo se presenta la solución adoptada para realizar el proyecto. En las siguientes subsecciones se describe la solución planteada, así como el proceso de desarrollo de esta.

3.1 Descripción de la solución

La solución propuesta consiste en la adaptación de la herramienta de denegación de servicio (DoS) llamada Tor's Hammer a un entorno virtual denominado Dorothea. Dorothea es una solución basada en Docker que permite crear topologías de red virtual para generar y recopilar datos de flujo. Su objetivo es simular tanto el tráfico de red benigno como el tráfico malicioso con el fin de obtener un conjunto de datos equilibrado para el entrenamiento y prueba de modelos de detección de ataques de denegación de servicio.

La adaptación de Tor's Hammer a Dorothea se realizó integrando su código en el nodo atacante de la herramienta. Se crearon dos tareas específicas en los nodos atacantes: "run_tasks_DDOS.py" para iniciar el ataque y "tasks_DDOS.py" para ejecutar el código de ataque utilizando Tor's Hammer.

Una vez implementada la herramienta Tor's Hammer en Dorothea, se procede a la recolección de los datos de flujo generados por los ataques. Se realizaron diferentes escenarios de ataques, desde un único atacante hasta diez atacantes coordinados, contra una única víctima. Además, se recopilaron datos de tráfico benigno para integrarlos en los conjuntos de datos, logrando un equilibrio entre ambos tipos de tráfico.

Posteriormente, se lleva a cabo el preprocesamiento de los datos, que incluye la limpieza de características, la reducción de dimensionalidad y la normalización de los datos. Estas etapas se realizan para mejorar el rendimiento de los modelos de detección y eliminar el sesgo de los datos generados.

Finalmente, se aplican diversas técnicas de aprendizaje automático, como Random Forest, Support Vector Machine, Decision Tree, Logistic Regression, Naive Bayes, K-Nearest Neighbors y XGBoost, para detectar ataques de denegación de

servicio. Se evalúa la efectividad de estas técnicas utilizando indicadores de rendimiento como la exactitud, precisión, exhaustividad y puntuación F1.

3.2 El proceso de desarrollo

3.2.1 Adaptación de la herramienta de DoS a Dorothea

En este punto se proporciona una explicación detallada sobre Dorothea, incluyendo la herramienta de denegación de servicio que se utilizó para obtener datos de flujo, así como los aspectos de su implementación

3.2.1.1. Dorothea [22]

Dorothea es una solución basada en Docker que permite crear topologías de red virtual para generar y recopilar datos de flujo. Utiliza un sensor de NetFlow para capturar los flujos generados por los paquetes que atraviesan una interfaz de red. Dorothea ofrece un entorno configurable y escalable que permite simular tanto el tráfico de red benigno como el malicioso.

Para generar tráfico benigno, Dorothea utiliza simuladores de tráfico de red que envían paquetes a una puerta de enlace. Los scripts personalizables permiten simular actividades legítimas realizadas por un usuario real, como navegación web, uso de correo electrónico o acceso remoto a escritorio. La generación de tráfico malicioso se distribuye entre diferentes nodos utilizando la biblioteca Python Celery, que permite la distribución de tareas.

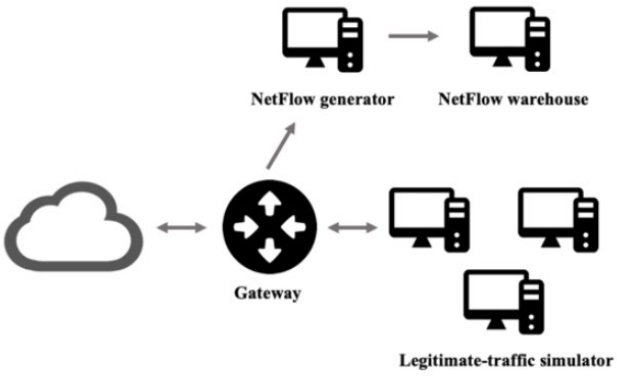

Figura 3.1. Esquema de tráfico normal

Fuente: https://seguridad.unileon.es/index.php/DOROTHEA

Una vez que se ejecuta la generación de tráfico malicioso, el nodo lanzador carga los scripts de ataque y los inserta en la cola de tareas. A partir de la cola de tareas, los nodos de ataque obtienen sus tareas y comienzan a ejecutar los ataques. Después de ejecutar todos los scripts de ataque y completar todas las tareas en la cola, Dorothea guarda los datos de flujo y apaga todos los nodos.

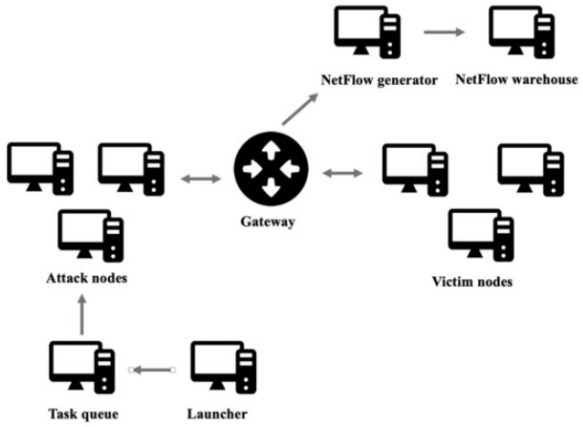

Figura 3.2 Esquema de trafico de ataque

Fuente: https://seguridad.unileon.es/index.php/DOROTHEA

Una característica destacada de Dorothea es que la generación de tráfico malicioso está aislada de Internet, lo que garantiza que todo el tráfico de red utilizado para construir datos de flujo corresponda a ataques maliciosos, facilitando así la etiquetación de datos.

3.2.1.2. Herramienta de denegación de servicio utilizada: Tor's Hammer [23]

Tor's Hammer es una herramienta de ataque DDoS de capa 7 diseñada para dirigirse a servidores web y aplicaciones. Su objetivo principal es inundar la aplicación y los servidores web con solicitudes HTTP POST lentas e incompletas. Esta técnica obliga a los subprocesos de conexión del servidor y de la aplicación a esperar la finalización de la solicitud, que nunca llega, provocando así una denegación de servicio.

Una de las características destacadas de Tor's Hammer es su capacidad para derribar un servidor web utilizando una sola máquina atacante. Esto se debe a su estrategia de enviar solicitudes incompletas a un ritmo lento, saturando la pila

TCP del servidor HTTP/S y agotando los recursos de manejo de conexión del servidor web y de la aplicación.

La detección de un ataque de Tor's Hammer puede ser difícil debido a su bajo volumen de tráfico, que a menudo está por debajo de los umbrales de detección basados en la tasa. Sin embargo, un aumento en el número de conexiones abiertas desde una máquina atacante puede ser un indicio de un ataque de Tor's Hammer en curso.

3.2.1.3. Implementación de Tor's Hammer en Dorothea

Para implementar Tor's Hammer dentro de Dorothea, se realizó un análisis exhaustivo del funcionamiento de Dorothea y se procedió a integrar el código de Tor's Hammer en el nodo atacante. Además, se creó un nodo esclavo, que actuaría como víctima el cual contendría un servidor Apache, y sería el receptor del ataque generado.

En el proceso de implementación, se desarrollaron dos nuevas tareas dentro de los nodos atacantes para llevar a cabo las siguientes acciones:

Tarea "run_tasks_DDOS.py":

Dentro de esta tarea, se inicia el ataque utilizando Tor's Hammer. Se configuran las direcciones IP y el puerto del objetivo seleccionado. En este caso, se utiliza la dirección IP específica 140.30.20.5 como la del nodo víctima y el puerto 80. Se establece un bucle para repetir el ataque dependiendo de los nodos atacantes que se hayan configurado en Dorothea, con un intervalo de espera de 10 segundos entre cada ataque. Esta configuración se puede ajustar según los requerimientos del escenario de prueba.

Tarea "tasks_DDOS.py":

En esta tarea se encuentra el código que ejecuta el ataque utilizando Tor's Hammer. Se utiliza el módulo os para ejecutar el comando necesario para lanzar el ataque con un límite de tiempo específico. Se construye un comando con la dirección IP y el puerto proporcionados, y se utiliza la función os.system() para ejecutar el comando con un límite de tiempo de 240 segundos. Se utiliza el comando sleep para esperar el tiempo límite y luego se envía una señal para

detener el proceso del ataque. Después de finalizar el ataque, se imprime un mensaje indicando que ha finalizado.

La implementación de Tor's Hammer en Dorothea se realizó mediante la creación de estas dos tareas específicas: "run_tasks_DDOS.py" para iniciar el ataque y "tasks_DDOS.py" para ejecutar el código de ataque utilizando Tor's Hammer. Estas tareas se configuran en los nodos atacantes de Dorothea, lo que permite simular ataques de denegación de servicio controlados y observar su impacto en la generación de datos de flujo. Estas tareas o código descrito se pueden observar en el repositorio que podemos encontrar en el anexo A. control de versiones.

3.2.2 Recolección de los datos

Una vez implementada la herramienta en Tor's Hammer, el paso siguiente fue ejecutar Dorothea para recopilar los datos del flujo generado por el ataque. Además, se recogieron datos benignos con el objetivo de integrarlos, consiguiendo así un conjunto de datos equitativamente dividido en un 50/50.

Dorothea ofrece la posibilidad de configurar el número de atacantes y víctimas, lo que me permitió realizar ataques de diversas formas. Entre ellos, realicé ataques de un único atacante contra una única víctima, de cinco atacantes contra una víctima y de diez atacantes contra una víctima. Opté por realizar los ataques contra una sola víctima, ya que, en un escenario de denegación de servicio, tiene más sentido que varios atacantes dirijan sus esfuerzos hacia un solo objetivo.

Asimismo, Dorothea permite configurar el muestreo (sampling), lo cual nos posibilita elegir uno de cada "x" paquetes. En este caso, recogí los datos sin muestreo y con un muestreo establecido en mil.

En las tablas 3.1 y 3.2, es posible apreciar los conjuntos de datos obtenidos a través de este proceso de recolección de datos. Es importante destacar que:

- El Dataset1 corresponde al conjunto de datos obtenido de un ataque perpetrado por un único atacante contra una única víctima.

- El Dataset2 representa los datos recopilados de un escenario en el que cinco atacantes lanzaron un ataque contra una única víctima.
- El Dataset3 contiene los datos provenientes de un caso en el que diez atacantes coordinaron un ataque contra una única víctima.

	Dataset1	**Dataset2**	**Dataset3**
Trafico normal	176.447	615.081	747.690
Trafico de ataque	176.447	615.081	747.690
Total	352.894	1.230.162	1.495.380

Tabla 3.1. Registros de los conjuntos de datos sin sampling

	Dataset1	**Dataset2**	**Dataset3**
Trafico normal	805	1.965	2.493
Trafico de ataque	805	1.965	2.493
Total	1.610	3.930	4.986

Tabla 3.2. Registros de los conjuntos de datos con sampling

3.2.3 El conjunto de datos, preprocesamiento y aprendizaje automático

Después de obtener el conjunto de datos, un paso fundamental antes de aplicar el aprendizaje automático es el preprocesamiento de las características que este contiene, las cuales se detallan en la Tabla 3.3. En cada conjunto de datos, podemos observar un total de 24 características.

Característica	Descripción
sysuptime	Milisegundos desde que se inició el dispositivo de exportación
unix secs	Recuento actual de segundos desde 0000 UTC 1970
unix nsecs	Nanosegundos residuales desde 0000 UTC 1970
engine type	Tipo de motor de conmutación de flujo
engine id	Número de ranura del motor de conmutación de flujo
exaddr	IP del exportador de flujo
srcaddr	Dirección IP de origen
dstaddr	Dirección IP de destino
nexthop	Dirección IP del siguiente router
input	Índice SNMP de la interfaz de entrada
output	Índice SNMP de la interfaz de salida
dpkts	Número de paquetes contenidos en el flujo
doctets	Número total de bytes de la capa 3 en los paquetes del flujo
first	Sysuptime al inicio del flujo
last	Sysuptime cuando se recibió el último paquete del flujo
srcport	Número de puerto de origen TCP/UDP
dstport	Número de puerto de destino TCP/UDP
tcp flags	Banderas TCP
prot	Tipo de protocolo IP (por ejemplo, TCP = 6; UDP = 17)
tos	Tipo de servicio IP (ToS)
src as	Número de sistema autónomo de origen, ya sea fuente o par
dst as	Número de sistema autónomo de destino, ya sea fuente o par
src mask	Bits de máscara de prefijo de dirección de origen
dst mask	Bits de máscara de prefijo de dirección de destino

Tabla 3.3 Características del dataset

El procesamiento de datos se ha llevado a cabo para mejorar el rendimiento de los modelos y eliminar el sesgo que puedan tener debido a la naturaleza de los datos generados.

Limpieza de características. En primer lugar, se convierten las direcciones IP en un valor numérico, y se verifica la existencia de columnas o filas vacías en los conjuntos de datos para evitar errores en la generación de los modelos.

Reducción de dimensionalidad. Esta técnica se utiliza para reducir la complejidad de los modelos. Dado que cada característica aumenta exponencialmente la complejidad de los modelos, lo que a su vez disminuye su capacidad de detección, se ha calculado la varianza de las características para disminuir su número. La varianza es una medida de dispersión que se utiliza para representar la variabilidad de un conjunto de datos con respecto a su media aritmética. Después de aplicar la varianza a las características de los flujos benignos y maliciosos, se han eliminado las características 'exaddr', 'engine type', 'engine id', 'src mask', 'dst mask', 'src as' y 'dst as', cuya varianza era 0. Además de las características mencionadas, se han eliminado las siguientes: 'unix secs', 'unix nsecs', 'sysuptime', 'first' y 'last'. Estas características están relacionadas con el tiempo y se han eliminado para evitar que los modelos estén sesgados dependiendo de cuándo se recogieron los datos. Finalmente, se ha eliminado la característica relacionada con las direcciones IP 'exaddr', 'srcaddr', 'dstaddr' y 'nexthop', debido a que el ataque se realiza de manera continua sobre una dirección IP, lo cual puede influir negativamente al aplicar aprendizaje automático.

Clasificación de los datos. Una vez realizado el paso anterior a los datos A los datos maliciosos se les asigna la etiqueta '1', mientras que a los datos no maliciosos se les asigna la etiqueta '0'. Esto se hace para diferenciar entre los datos maliciosos y no maliciosos durante el entrenamiento y la prueba. Después de realizar esto se mezclan los datos de manera aleatoria formando un solo dataset.

División de los datos en conjuntos de entrenamiento y prueba. Una vez que los datos están combinados y mezclados, se dividen en conjuntos de entrenamiento y prueba. En este proyecto el conjunto de datos de prueba constituirá el 20% del conjunto de datos total.

Normalización de Datos. Es necesario normalizar el valor de las características a un rango específico para analizar los datos con precisión. Esta

técnica se realiza para evitar errores relacionados con la escala de los datos. Por ejemplo, el tamaño de una característica puede ser mayor que otras, lo que puede generar un sesgo al interpretar que una característica tiene más peso que otra.

En este trabajo, se ha utilizado la técnica de normalización de datos lineal basada en el min-max. La normalización min-max se muestra en la Figura donde los X's son los valores para normalizar y Min(x) y Max(x) son los valores mínimo y máximo de las características antes de la normalización. Después de aplicar esta normalización, todos los datos que conforman los datasets están en una escala entre '0' y '1'.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Figura 3.3. Ecuación Min-Max

Fuente: https://androidkt.com/how-to-scale-data-to-range-using-minmax-normalization/

Una vez finalizado el preprocesamiento, se aplican las técnicas de aprendizaje automático para detectar ataques de denegación de servicio. Estas técnicas se seleccionaron después de realizar una revisión sistemática de la literatura sobre la detección de ataques de denegación de servicio, la cual se encuentra en el punto 1.4 "Estado de la cuestión". De entre las técnicas identificadas, se eligieron aquellas que son más ampliamente utilizadas y que presentan una mayor tasa de exactitud, que son las siguientes: Random Forest (RF), Support Vector Machine (SVM), Decision Tree (DT), Logistic Regression (LR), Naive Bayes (NB), K-Nearest Neighbors (KNN) y XGBoost (XGB). Además, como último paso, se empleará un "ensemble classifier" que utilizará los modelos mencionados previamente, con el objetivo de combinar las predicciones de todos los modelos para mejorar la precisión y reducir el riesgo de sobreajuste.

Después de entrenar los modelos con los datos de entrenamiento, se evalúa la efectividad de estas técnicas calculando varios indicadores clave de rendimiento (KPI), que incluyen Accuracy (Exactitud), Precision (Precisión), Recall

(Exhaustividad) y F1 Score, utilizando los datos de prueba. Los resultados obtenidos de estas evaluaciones se presentarán en el Capítulo 4.

3.2.4. Herramientas utilizadas

Durante el proceso de desarrollo, se han utilizado diversas herramientas que han sido fundamentales para la implementación, análisis y evaluación de la solución. A continuación, se describen las principales herramientas utilizadas:

- **Dorothea**: Es una solución basada en Docker que se ha empleado para crear topologías de red virtual y generar/recopilar datos de flujo. Dorothea ha proporcionado un entorno configurable y escalable para simular tanto el tráfico benigno como el tráfico malicioso en los escenarios de ataques de denegación de servicio.
- **Tor's Hammer:** Esta herramienta de ataque DDoS de capa 7 ha sido adaptada e integrada en Dorothea para llevar a cabo los ataques en los nodos atacantes. Tor's Hammer se ha utilizado para inundar los servidores web y aplicaciones con solicitudes HTTP POST lentas e incompletas, generando una denegación de servicio. El código de Tor's Hammer implementado en Dorothea se obtuvo del repositorio que podemos ver en [24]
- **Git**: Se ha utilizado el sistema de control de versiones Git para gestionar el código fuente y mantener un registro de los cambios realizados en el desarrollo del proyecto.
- **Excel**: Microsoft Excel ha sido utilizado como una herramienta de visualización de los datos generados por dorothea.
- **Python**: Python ha sido el lenguaje de programación principal utilizado en el desarrollo de la solución. Se han empleado diversas bibliotecas y módulos de Python para el preprocesamiento de datos, la implementación de los modelos de aprendizaje automático y la visualización de resultados.
- **Scikit-learn (Sklearn):** Esta biblioteca de Python se ha utilizado para implementar y entrenar los modelos de aprendizaje automático, como Random Forest, Support Vector Machine, Decision Tree, Logistic Regression, Naive Bayes, K-Nearest Neighbors, entre otros. Sklearn ha

proporcionado herramientas eficientes para el procesamiento de datos y la evaluación de los modelos.

- **Xgboost**: Xgboost es una biblioteca de Python para la implementación de algoritmos de boosting. Se ha utilizado en este proyecto para entrenar modelos utilizando el algoritmo de boosting XGBoost.
- **Pandas**: Pandas es una biblioteca de Python que se ha empleado para el manejo y manipulación de los datos en forma de DataFrames. Pandas ha facilitado las tareas de limpieza, transformación y filtrado de los conjuntos de datos.
- **Matplotlib**: Esta biblioteca de visualización de datos en Python ha sido utilizada para crear gráficos y visualizaciones que ayudan a comprender y presentar los resultados obtenidos.

Capítulo 4: Evaluación

Este capítulo demuestra la validez de la solución propuesta. En primer lugar, se hablará sobre cómo se ha realizado el proceso de evaluación y más tarde se analizarán los resultados obtenidos.

4.1 Proceso de evaluación

Para validar la solución propuesta, se han empleado diversos indicadores clave de rendimiento (KPI) que evalúan el desempeño del modelo. Estos indicadores incluyen:[25]

Accuracy (Exactitud): Es una medida que evalúa qué tan acertado es el modelo al clasificar correctamente todas las muestras, ya sean positivas o negativas. Un valor cercano a 1 indica un mejor rendimiento del modelo. Podemos ver la fórmula utilizada a continuación:

$$Accuracy = \frac{Verdaderos\ Positivos + Verdaderos\ Negativos}{Total\ de\ Muestras}$$

Precision (Precisión): Esta mide la proporción de muestras clasificadas como positivas que realmente lo son en relación con todas las clasificadas como positivas. Una alta precisión sugiere que el modelo presenta una baja tasa de falsos positivos. La fórmula correspondiente es:

$$Precision = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos\ + Falsos\ Positivos}$$

Recall (Exhaustividad): Evalúa la proporción de muestras verdaderamente positivas que el modelo identifica correctamente. Un alto recall señala que el modelo presenta una baja tasa de falsos negativos. La fórmula se presenta a continuación:

$$Recall = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos\ + Falsos\ Negativos}$$

F1 Score: Es una métrica que combina precisión y exhaustividad. Su valor varía entre 0 y 1, siendo 1 el rendimiento óptimo. Su fórmula es la siguiente:

$$F1\ Score = 2\ \times \frac{Precision\ \times Recall}{Precision\ \times Recall}$$

Adicionalmente a estos KPI, se consideraron los siguientes indicadores:

- **Tiempo de entrenamiento**: El tiempo de entrenamiento es el tiempo que toma al modelo aprender de los datos de entrenamiento y ajustar sus parámetros. Es relevante para evaluar la eficiencia del modelo durante la fase de entrenamiento.
- **Tiempo de predicción**: El tiempo de predicción es el tiempo que toma al modelo hacer predicciones para nuevas muestras una vez que ha sido entrenado. Es importante medir la velocidad de predicción, especialmente en aplicaciones que requieren respuestas rápidas y en tiempo real.

Al analizar todos estos KPI en conjunto se podrá llegar a una conclusión analizando los resultados.

4.2 Análisis y resultados

Este subapartado se dividirá en dos secciones, y en ambas se emplearán tablas y gráficos para ilustrar los resultados obtenidos de los distintos conjuntos de datos presentados el apartado 3.2.2. La primera sección detallará los resultados al aplicar los indicadores clave de rendimiento en los datasets sin haber utilizado "sampling". En la segunda, se presentarán los resultados correspondientes a los datasets donde se implementó el "sampling", Una vez presentados los resultados obtenidos se procederá a realizar un análisis de los resultados obtenidos.

4.2.1. Conjuntos de datos sin muestreo (Sampling)

4.2.1.1. Dataset1

En el Dataset1, que contiene un total de 352,894 datos y corresponde al conjunto de datos obtenido de un ataque realizado por un único atacante a una única víctima, los resultados obtenidos al aplicar los indicadores clave de rendimiento (KPI) descritos en el apartado 4.1 para cada uno de los modelos entenados se presentan en la Tabla 4.1. Además, en la Figura 4.1 se pueden apreciar los tiempos de entrenamiento y predicción necesarios para obtener dichos resultados.

Algoritmo	Accuracy	Recall	Precision	F1 Score
KNN	99,78%	99,68%	99,88%	99,78%
LR	85.98%	74.53%	96,72%	84,19%
SVC	86%	75%	97%	84%
RF	100%	100%	100%	100%
DT	100%	100%	100%	100%
XGB	100%	100%	100%	100%
NB	65,02%	100%	58,88%	74,12%
Ensemble	99,99%	100%	99,99%	99,99%

Tabla 4.1. Resultados de los KPI del Dataset1

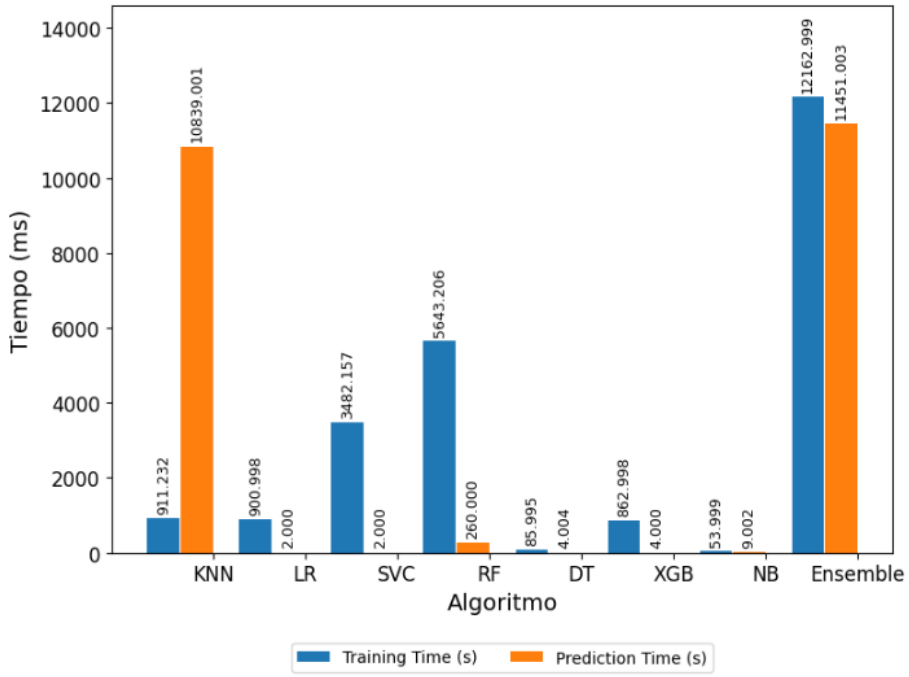

Figura 4.1. Gráfico Tiempos Dataset1

Fuente: Elaboración propia

4.2.1.2. Dataset2

En el Dataset2, que abarca un total de 1,230,162 datos y corresponde al conjunto de datos obtenido de un ataque realizado por cinco atacantes a una única víctima, los resultados de los indicadores clave de rendimiento (KPI) mencionados en el apartado 4.1 para cada uno de los modelos entenados se detallan en la Tabla 4.2. Asimismo, en la Figura 4.2 se representan los tiempos de entrenamiento y predicción necesarios para obtener dichos resultados.

Algoritmo	Accuracy	Recall	Precision	F1 Score
KNN	99,93%	99,91%	99,96%	99,93%
LR	94,08%	90,79%	97,20%	93,88%
SVC	94,11%	90,82%	97,21%	93,91%
RF	100,00%	100,00%	100,00%	100,00%
DT	100,00%	100,00%	100,00%	100,00%
XGB	100,00%	100,00%	100,00%	100,00%
NB	77,96%	100,00%	69,41%	81,94%
Ensemble	100,00%	100,00%	100,00%	100,00%

Tabla 4.2. Resultados de los KPI del Dataset2

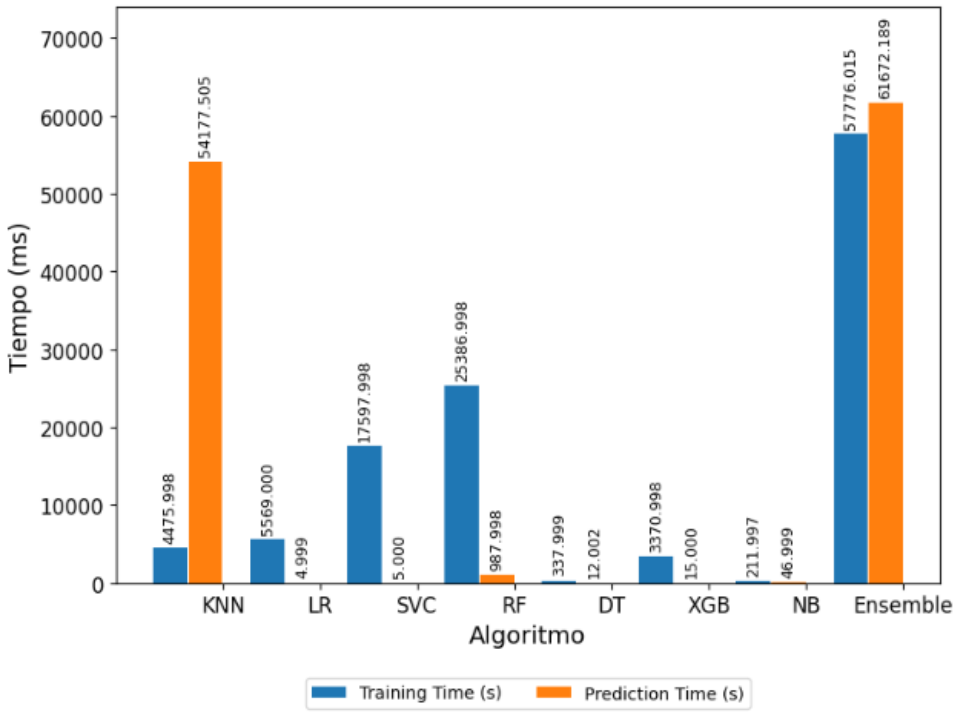

Figura 4.2. Gráfico Tiempos Dataset2

Fuente: Elaboración propia

4.2.1.3. Dataset3

El Dataset3, que engloba un conjunto de datos aún más amplio, con un total de 1.495.380 registros, representa una situación especial en la que se simularon ataques llevados a cabo por 10 atacantes contra una única víctima. Los resultados de los indicadores clave de rendimiento (KPI) descritos en el apartado 4.1 para cada uno de los modelos entenados se detallan en la Tabla 4.3. Además, en la Figura 4.3 se presentan de manera gráfica los tiempos de entrenamiento y predicción necesarios para obtener estos resultados.

Algoritmo	Accuracy	Recall	Precision	F1 Score
KNN	99,96%	99,94%	99,98%	99,96%
LR	91,68%	86,07%	96,91%	91,17%
SVC	91,70%	86,07%	96,94%	91,18%
RF	100,00%	100,00%	100,00%	100,00%
DT	100,00%	100,00%	100,00%	100,00%
XGB	100,00%	100,00%	100,00%	100,00%
NB	69,01%	100,00%	61,68%	76,30%
Ensemble	100,00%	100,00%	100,00%	100,00%

Tabla 4.3. Resultados de los KPI del Dataset3

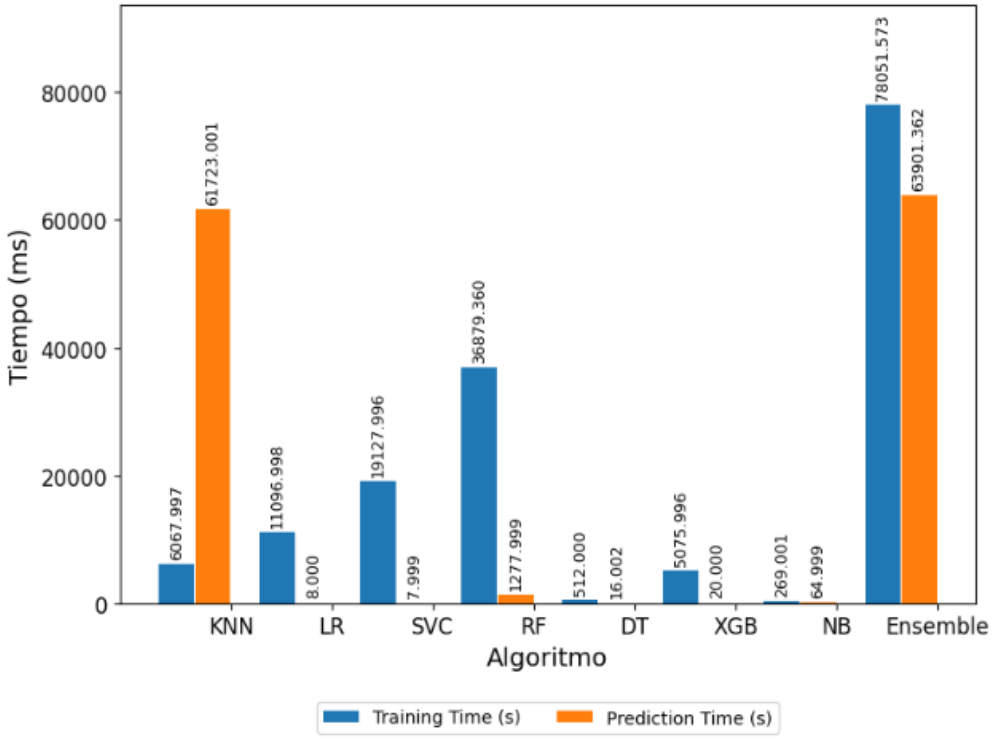

Figura 4.3. Gráfico Tiempos Dataset3

Fuente: Elaboración propia

4.2.1.4. Análisis de los resultados

En las tablas presentadas, se observa un comportamiento consistente en la mayoría de ellas. Los algoritmos que han mostrado mejores resultados son Random Forest (RF), Decision Tree (DT), eXtreme Gradient Boosting (XGBoost) y k-Nearest Neighbors (KNN), todos con una precisión superior al 99%. Por su parte, algoritmos como Logistic Regression (LR) y Support Vector Classifier (SVC) presentaron una precisión ligeramente inferior, aunque aún alta, superando el 86%. En contraste, Naive Bayes (NB) ha reportado los resultados menos destacados, con una precisión que oscila entre el 65% y el 78%.

Con respecto a los tiempos de entrenamiento, es destacable que los algoritmos Random Forest y Support Vector Classifier son los que requieren más tiempo, aunque no superan los 37 segundos.

En lo que respecta a los tiempos de predicción, podemos destacar a k-Nearest Neighbors (KNN) por tener un tiempo considerablemente mayor en comparación con el resto de los algoritmos aplicados. Esto es comprensible, ya que, durante la predicción, KNN debe recorrer todo el conjunto de datos para encontrar los 'k' vecinos más cercanos, lo cual puede resultar computacionalmente costoso, especialmente con conjuntos de datos extensos.

Finalmente, cabe destacar que, al integrar estos algoritmos para formar un "ensemble classifier", se alcanza una precisión que supera el 99.99% en todos los datasets mostrados anteriormente. Es importante mencionar que los tiempos de predicción y entrenamiento del "ensemble classifier" resultan ser superiores, ya que se requiere aplicar todos los algoritmos anteriores, comparado con el tiempo que tomaría cualquier algoritmo individualmente

4.2.2. Conjuntos de datos con muestreo (Sampling)

4.2.2.1. Dataset1

En el Dataset1 obtenido con sampling, que contiene un total de 1.610 datos y corresponde al conjunto de datos obtenido de un ataque realizado por un único atacante a una única víctima, los resultados obtenidos al aplicar los indicadores clave de rendimiento (KPI) descritos en el apartado 4.1 para cada uno de los modelos entenados se presentan en la Tabla 4.4. Además, en la Figura 4.4 se pueden apreciar los tiempos de entrenamiento y predicción necesarios para obtener dichos resultados.

Algoritmo	Accuracy	Recall	Precision	F1 Score
KNN	97,52%	96,86%	98,09%	97,47%
LR	84,16%	70,44%	96,55%	81,45%
SVC	88,51%	79,25%	96,92%	87,20%
RF	100,00%	100,00%	100,00%	100,00%
DT	100,00%	100,00%	100,00%	100,00%
XGB	100,00%	100,00%	100,00%	100,00%
NB	79,50%	100,00%	70,67%	82,81%
Ensemble	99,07%	100,00%	98,15%	99,07%

Tabla 4.4. Resultados de los KPI del Dataset1 con Sampling

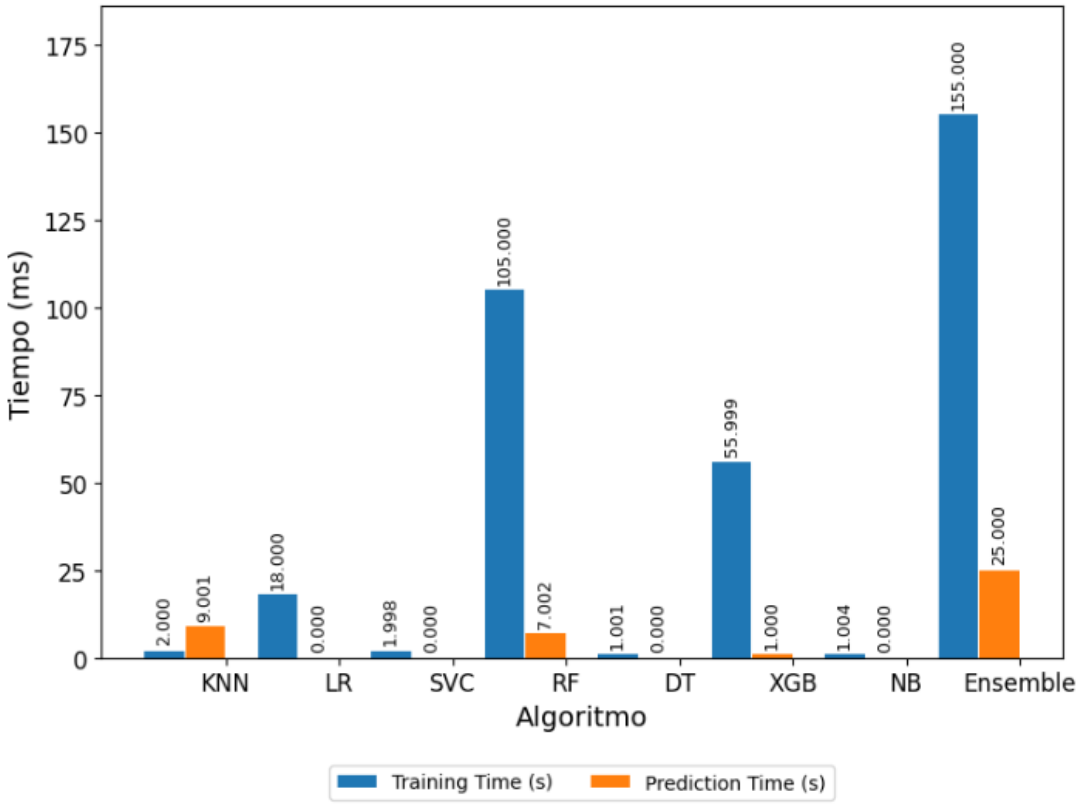

Figura 4.4. Gráfico Tiempos Dataset1 con Sampling

Fuente: Elaboración propia

4.2.2.2. Dataset2

En el Dataset2 obtenido con sampling, que abarca un total de 3.930 datos y corresponde al conjunto de datos obtenido de un ataque realizado por cinco atacantes a una única víctima, los resultados de los indicadores clave de rendimiento (KPI) mencionados en el apartado 4.1 para cada uno de los modelos entenados se detallan en la Tabla 4.2. Asimismo, en la Figura 4.2 se representan los tiempos de entrenamiento y predicción necesarios para obtener dichos resultados.

Algoritmo	Accuracy	Recall	Precision	F1 Score
KNN	96,18%	96,70%	95,73%	96,21%
LR	86,90%	76,65%	96,49%	85,43%
SVC	86,90%	76,65%	96,49%	85,43%
RF	100,00%	100,00%	100,00%	100,00%
DT	100,00%	100,00%	100,00%	100,00%
XGB	100,00%	100,00%	100,00%	100,00%
NB	76,59%	99,49%	68,29%	80,99%
Ensemble	98,73%	99,49%	98,00%	98,74%

Tabla 4.5. Resultados de los KPI del Dataset2 con Sampling

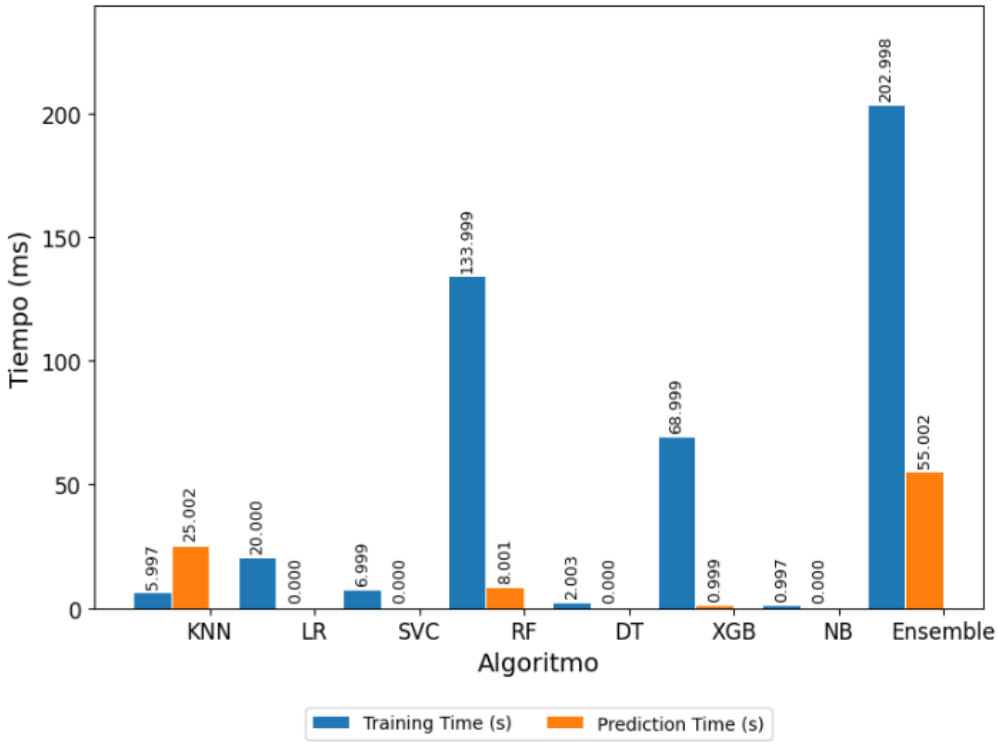

Figura 4.5. Gráfico Tiempos Dataset2 con Sampling

Fuente: Elaboración propia

4.2.2.2. Dataset3

El Dataset3 obtenido con sampling, con un total de 4.986 registros, representa una situación especial en la que se simularon ataques llevados a cabo por 10 atacantes contra una única víctima. Los resultados de los indicadores clave de rendimiento (KPI) mencionados en el apartado 4.1 para cada uno de los modelos entenados se detallan en la Tabla 4.3. Además, en la Figura 4.3 se presentan de manera gráfica los tiempos de entrenamiento y predicción necesarios para obtener estos resultados.

Algoritmo	Accuracy	Recall	Precision	F1 Score
KNN	95,79%	96,48%	95,36%	95,91%
LR	86,67%	78,28%	94,79%	85,74%
SVC	86,77%	78,47%	94,80%	85,87%
RF	100,00%	100,00%	100,00%	100,00%
DT	100,00%	100,00%	100,00%	100,00%
XGB	100,00%	100,00%	100,00%	100,00%
NB	81,36%	99,80%	73,38%	84,58%
Ensemble	98,50%	99,80%	97,33%	98,55%

Tabla 4.6. Resultados de los KPI del Dataset3 con Sampling

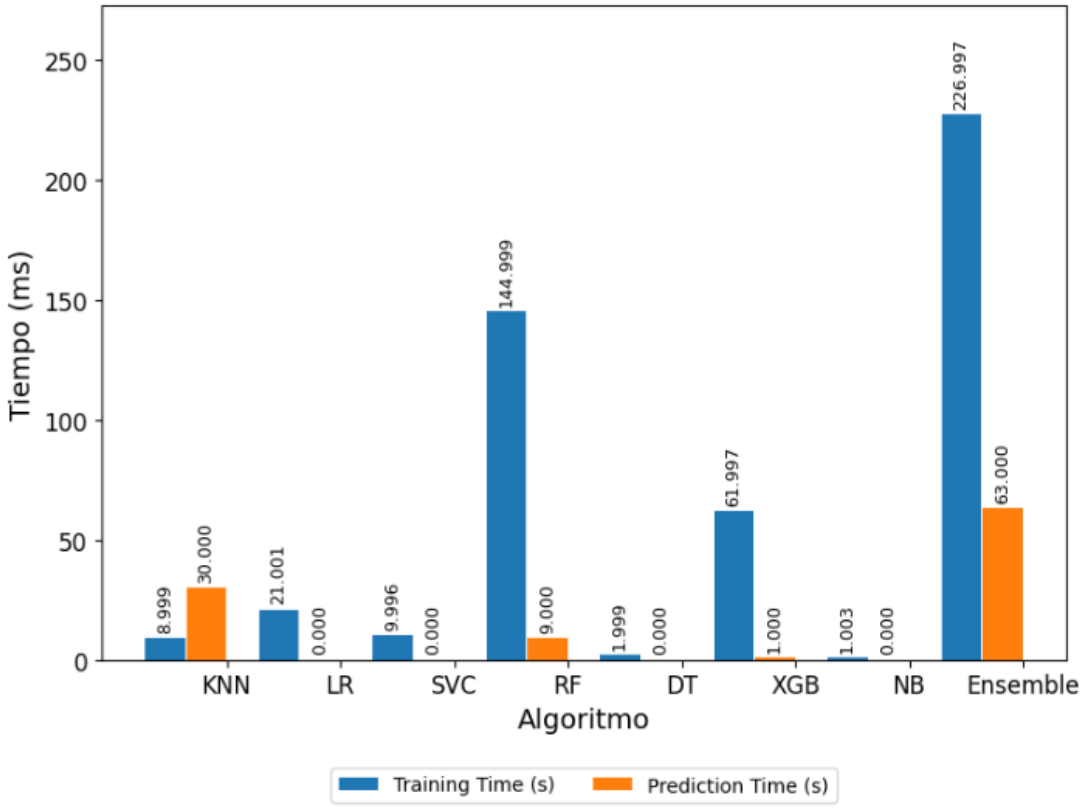

Figura 4.6. Gráfico Tiempos Dataset3 con Sampling

Fuente: Elaboración propia

4.2.1.4. Análisis de los resultados

Los resultados obtenidos del dataset con muestreo, los cuales son prácticamente iguales a los obtenidos en el apartado anterior. Los modelos Random Forest (RF), Decision Tree (DT) y eXtreme Gradient Boosting (XGBoost) alcanzan una exactitud del 100%. No obstante, para k-Nearest Neighbors (KNN) se observa una ligera disminución en la exactitud debido a tener menos datos en el conjunto, aunque sigue siendo alta, alcanzando entre un 95% y un 88% de exactitud. Por otro lado, los modelos Logistic Regression (LR) y Support Vector Classifier (SVC) Support Vector Classifier (SVC) también mantienen una exactitud similar a la obtenida sin muestreo, rondando entre un 86% y un 88%. En cuanto a Naive Bayes, los resultados siguen siendo los peores al igual que en el caso sin muestreo, con una exactitud en torno al 75% y 80%.

En relación con los tiempos, al tratarse de conjuntos de datos más pequeños, los tiempos de entrenamiento y predicción son excepcionalmente bajos, sin superar el segundo.

Respecto al ensemble clasificador, se ha obtenido una exactitud menor que la obtenida sin muestreo, aunque sigue siendo mayor al 98%.

Capítulo 5: Conclusión

En este proyecto, se ha realizado a cabo un análisis exhaustivo de denegaciones de servicio en flujos de datos, empleando técnicas de aprendizaje automático para su detección en redes de área extensa mediante el análisis de los datos de flujo de los routers core de internet. Los objetivos planteados al inicio de esta investigación se han alcanzado satisfactoriamente, y se han cumplido los siguientes objetivos específicos:

Análisis del Estado Actual del Problema: Se ha realizado una revisión de la literatura y un estudio general del problema, identificando los algoritmos de aprendizaje automático más eficaces utilizados en la detección de denegaciones de servicio en flujos de datos.

Investigación de Herramientas para Simulación: Se ha investigado y evaluado una serie de herramientas disponibles para la simulación de denegaciones de servicio, seleccionando la más adecuada para nuestro proyecto.

Implementación de la Herramienta en Dorothea: La herramienta seleccionada se ha implementado en el entorno de Dorothea, asegurando su compatibilidad y funcionalidad.

Recolección de Datos Generados por Dorothea: Se ha llevado a cabo la recopilación de datos generados a través de Dorothea durante las simulaciones de denegaciones de servicio.

Preprocesamiento de Datos: Se ha realizado un riguroso preprocesamiento de los datos recolectados preparándolos para su posterior análisis.

Aplicación de Algoritmos de Aprendizaje Automático: Se han aplicado diversos algoritmos de aprendizaje automático sobre los datos preprocesados con el fin de detectar denegaciones de servicio de manera eficiente.

Análisis de Resultados: Se ha llevado a cabo un análisis minucioso de los resultados obtenidos, evaluando el rendimiento de los algoritmos de aprendizaje automático en la detección de denegaciones de servicio en flujos de datos.

5.1 Aportaciones realizadas

El proyecto desarrollado ha realizado las siguientes aportaciones:

Aportación 1: Se llevó a cabo un estudio del problema de las Denegaciones de servicio, en el cual se abordó el concepto de la amenaza, su contexto y sus características generales. Este estudio se detalló en el Capítulo 1, donde se realizó un análisis centrado en el concepto de las denegaciones de servicio.

Aportación 2: Se realizó una revisión de la literatura que abordó las últimas tendencias en la aplicación de técnicas de aprendizaje automático para la detección de denegaciones de servicio en el tráfico de red. Este estudio proporcionó un resumen de los algoritmos de aprendizaje automático, conjuntos de datos y herramientas más utilizadas en la aplicación de técnicas de machine learning en el ámbito del tráfico de red.

Aportación 3: Se llevó a cabo la creación de distintos flujos de datos obtenidos a través de DOROTHEA, implementando una herramienta llamada "Thorshammer" para simular denegaciones de servicio.

Aportación 4: Se desarrolló un cuaderno de Jupyter desde el cual se puede realizar el preprocesamiento de los distintos datasets, la aplicación de técnicas de aprendizaje automático y el análisis de resultados mediante indicadores de rendimiento.

En conclusión, tras finalizar el estudio, se confirma que se ha logrado el objetivo principal: realizar un análisis de las denegaciones de servicio en un flujo de datos utilizando el aprendizaje automático para su detección.

5.2 Trabajos Futuros

A pesar de los logros alcanzados en este proyecto, existen diversas oportunidades para mejorar y ampliar esta investigación en el futuro. A continuación, se enumeran posibles mejoras y líneas futuras que pueden aplicarse al estudio:

- **Diversificación de Conjuntos de Datos:** Se podría explorar la utilización de conjuntos de datos adicionales obtenidos mediante diferentes herramientas o técnicas de denegación de servicio en lugar de limitarse al conjunto de

datos actual centrado en Torhammer. Esto permitiría una evaluación más completa de la capacidad de detección de ataques.

- **Gestión del Sobreajuste y Subajuste:** Para mejorar la robustez de los modelos de aprendizaje automático, se pueden emplear herramientas y técnicas específicas para evaluar y mitigar el sobreajuste (overfitting) o subajuste (underfitting) de los resultados obtenidos. Esto garantizaría un rendimiento más confiable en situaciones diversas.
- **Optimización de Hiperparámetros:** Sería beneficioso utilizar herramientas de búsqueda de hiperparámetros para encontrar configuraciones óptimas para cada algoritmo de aprendizaje automático utilizado en el proyecto. Esto podría aumentar significativamente la precisión de la detección de ataques de denegación de servicio.
- **Implementación del "Ensemble Classifier":** Una línea futura de investigación prometedora sería la implementación de un "ensemble classifier" en un sistema en tiempo real. Esto permitiría combinar múltiples modelos de detección para mejorar aún más la capacidad de identificar ataques de denegación de servicio de manera efectiva.

Es importante destacar que, aunque este proyecto se ha enfocado principalmente en la detección de ataques de denegación de servicio utilizando la herramienta Torhammer, no podemos garantizar que los modelos desarrollados sean igualmente efectivos en la detección de otros tipos de ataques. Esta limitación resalta la necesidad de investigaciones adicionales y especializadas para abordar otros tipos de amenazas en el ámbito de la seguridad de la red.

5.3 Problemas encontrados

- **Problema:** Docker no pudo montar Dorothea correctamente.
    - **Solución:** Se descubrió que el problema era debido a que Dorothea se estaba ejecutando en Ubuntu 22, una versión que no es compatible con Dorothea debido a diferencias en las cabeceras. La solución fue migrar a una versión anterior de Ubuntu compatible con Dorothea, lo que requirió varias semanas hasta dar con la solución.

- **Problema:** Al intentar realizar varios ataques con Dorothea, solo se efectuaba uno.
    - **Solución:** Mediante la revisión del código, se identificó que solamente un ataque se añadía a la cola. Se implementó un bucle para permitir la ejecución de múltiples ataques consecutivos, esta solución implico realizar cambios en el código lo que llevo aproximadamente una semana.
- **Problema:** Falta de conocimiento sobre aprendizaje automático.
    - **Solución:** Se llevaron a cabo investigaciones y capacitaciones sobre las tecnologías y conceptos relevantes para adquirir una comprensión adecuada sobre el aprendizaje automático. Esta solución me permitió conocer un campo en el que apenas había profundizado, para adquirir los conocimientos necesarios necesite dos semanas.
- **Problema:** Al ejecutar ataques con más de 10 atacantes, el sistema se volvía inestable y se producía un bloqueo.
    - **Solución:** Dadas las restricciones de hardware del equipo utilizado, se decidió limitar la ejecución a no más de 10 atacantes simultáneos. La decisión de fue tomada después de una reunión teniendo en cuenta que el hardware desde el que se estaba ejecutando no podía con la carga de trabajo.
- **Problema:** La aplicación del método grid search para ajustar los hiperparámetros en modelos de aprendizaje automático tomaba un tiempo excesivo.
    - **Solución:** Debido al tiempo excesivo que requería el método grid search se decidió descartar su uso para la optimización de hiperparámetros.

5.4 Opiniones personales

El desarrollo de este proyecto me ha permitido ampliar mis conocimientos sobre las denegaciones de servicio, aprendiendo cómo se llevan a cabo y qué tipos de denegación existen, así como las herramientas que se pueden utilizar para ejecutarlos. También me ha permitido aprender y desarrollar habilidades en el entorno Docker, ampliando mis conocimientos previos. Además, he descubierto la herramienta DOROTHEA, la cual no conocía, que se utiliza para generar los datos de flujo que posteriormente se analizaron.

Asimismo, he tenido la oportunidad de aprender sobre el aprendizaje automático y sus diferentes algoritmos y técnicas, lo que me ha permitido adquirir habilidades en el tratamiento de datos generados para aplicar estas técnicas de aprendizaje automático.

Bibliografía

[1] B. A. Kitchenham, D. (David) Budgen, y P. Brereton, *Evidence-based software engineering and systematic reviews*. Accedido: 9 de julio de 2023. [En línea]. Disponible en: https://www.routledge.com/Evidence-Based-Software-Engineering-and-Systematic-Reviews/Kitchenham-Budgen-Brereton/p/book/9780367575335

[2] J. Garzas, *Gestion Agil de Proyectos Software*, Kybele Consulting., vol. 1st ed. 2012.

[3] «Explicando 12 tipos comunes de ataques DDoS | EasyDMARC». https://easydmarc.com/blog/es/explicando-12-tipos-comunes-de-ataques-ddos/ (accedido 18 de julio de 2023).

[4] «Amazon AWS dice haber mitigado el mayor ataque DDoS de la historia: un incidente que causó un tráfico de 2,3 Tbps». https://www.xataka.com/seguridad/amazon-aws-dice-haber-mitigado-mayor-ataque-ddos-historia-incidente-que-causo-trafico-2-3-tpbs (accedido 8 de julio de 2023).

[5] «GitHub acaba de sobrevivir el ataque DDoS más grande de la historia». https://www.genbeta.com/actualidad/github-acaba-de-sobrevivir-el-ataque-ddos-mas-grande-de-la-historia (accedido 8 de julio de 2023).

[6] «Así se gestó el ciberataque más grave de los últimos 10 años | Tecnología Home | EL MUNDO». https://www.elmundo.es/tecnologia/2016/10/22/580b10e5268e3e06158b45e0.html (accedido 8 de julio de 2023).

[7] D. Moher, «Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement», *Ann Intern Med*, vol. 151, n.º 4, p. 264, ago. 2009, doi: 10.7326/0003-4819-151-4-200908180-00135.

[8] A. Roehrs, C. A. Da Costa, R. Da Rosa Righi, y K. S. F. De Oliveira, «Personal Health Records: A Systematic Literature Review», *J Med Internet Res*

2017;19(1):e13 https://www.jmir.org/2017/1/e13, vol. 19, n.º 1, p. e5876, ene. 2017, doi: 10.2196/JMIR.5876.

[9] C. Schardt, M. B. Adams, T. Owens, S. Keitz, y P. Fontelo, «Utilization of the PICO framework to improve searching PubMed for clinical questions», *BMC Med Inform Decis Mak*, vol. 7, 2007, doi: 10.1186/1472-6947-7-16.

[10] A. H. Azizan *et al.*, «A machine learning approach for improving the performance of network intrusion detection systems», *Annals of Emerging Technologies in Computing*, vol. 5, n.º Special issue 5, pp. 201-208, 2021, doi: 10.33166/AETiC.2021.05.025.

[11] M. Gohil y S. Kumar, «Evaluation of Classification algorithms for Distributed Denial of Service Attack Detection», en *Proceedings - 2020 IEEE 3rd International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2020*, Institute of Electrical and Electronics Engineers Inc., dic. 2020, pp. 138-141. doi: 10.1109/AIKE48582.2020.00028.

[12] Ismail *et al.*, «A Machine Learning-Based Classification and Prediction Technique for DDoS Attacks», *IEEE Access*, vol. 10, pp. 21443-21454, 2022, doi: 10.1109/ACCESS.2022.3152577.

[13] S. A. M. Al-Juboori, F. Hazzaa, Z. S. Jabbar, S. Salih, y H. M. Gheni, «Man-in-the-middle and denial of service attacks detection using machine learning algorithms», *Bulletin of Electrical Engineering and Informatics*, vol. 12, n.º 1, pp. 418-426, feb. 2023, doi: 10.11591/eei.v12i1.4555.

[14] M. I. Kareem y M. N. Jasim, «Fast and accurate classifying model for denial-of-service attacks by using machine learning», *Bulletin of Electrical Engineering and Informatics*, vol. 11, n.º 3, pp. 1742-1751, jun. 2022, doi: 10.11591/eei.v11i3.3688.

[15] T. Aytaç, M. A. Aydın, y A. H. Zaim, «Detection DDOS attacks using machine learning methods», *Electrica*, vol. 20, n.º 2, pp. 159-167, jun. 2020, doi: 10.5152/electrica.2020.20049.

[16] «Sueldo de Scrum master/a en España». https://es.indeed.com/career/scrum-master/salaries (accedido 19 de julio de 2023).

[17] «Sueldo de Analista de seguridad informática en España». https://es.indeed.com/career/analista-de-seguridad-inform%C3%A1tica/salaries (accedido 19 de julio de 2023).

[18] «Sueldo de Analistas de datos en España». https://es.indeed.com/career/analistas-de-datos/salaries (accedido 19 de julio de 2023).

[19] «BOE-A-1996-8930 Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia.» https://www.boe.es/buscar/act.php?id=BOE-A-1996-8930 (accedido 6 de septiembre de 2023).

[20] «REGLAMENTO (UE) 2016/ 679 DEL PARLAMENTO EUROPEO Y DEL CONSEJO - de 27 de abril de 2016 - relativo a la protección de las personas físicas en l o que respecta al tratamiento de datos personales y a la libre circu lación de estos datos y por el que se deroga la Directiva 95/ 46/ CE (Reglamento general de protección de datos)».

[21] «BOE-A-2021-1192 Real Decreto 43/2021, de 26 de enero, por el que se desarrolla el Real Decreto-ley 12/2018, de 7 de septiembre, de seguridad de las redes y sistemas de información.» https://www.boe.es/diario_boe/txt.php?id=BOE-A-2021-1192 (accedido 6 de septiembre de 2023).

[22] «DOROTHEA - Security Unileon». https://seguridad.unileon.es/index.php/DOROTHEA (accedido 18 de julio de 2023).

[23] «Tor's Hammer Attack | Perform DDoS Attack Using Tors Hammer | MazeBolt Knowledge Base». https://kb.mazebolt.com/knowledgebase/tors-hammer-attack/ (accedido 18 de julio de 2023).

[24] «Karlheinzniebuhr/torshammer: A DOS Attack Tool». https://github.com/Karlheinzniebuhr/torshammer (accedido 18 de julio de 2023).

[25] «Selección de métricas para los modelos de aprendizaje automático | Fayrix». https://fayrix.com/machine-learning-metrics_es (accedido 21 de julio de 2023).

Anexo A: Seguimiento de proyecto fin de carrera

En la Figura A.1. podemos observar a la izquierda las diferentes tareas realizadas en este proyecto con el tiempo establecido para realizarlas. Además, en la parte derecha, se muestra el diagrama de Gantt correspondiente.

Id	Nombre de tarea	Comienzo	Fin
1	inicio del proyecto	jue 22/12/22	jue 22/12/22
2	Reunion de planificacion del Sprint 1	jue 22/12/22	jue 22/12/22
3	**Investigación (Sprint 1)**	**vie 23/12/22**	**jue 02/02/23**
4	Investigación sobre el estado actual de las denegaciones de servicio	vie 23/12/22	mié 11/01/23
5	Búsqueda detallada de herramientas para ataques DoS.	jue 12/01/23	jue 19/01/23
6	Estudio sobre la aplicación de aprendizaje automático en la detección de denegaciones de servicio.	vie 20/01/23	mar 31/01/23
7	Identificación de los mejores algoritmos y enfoques actuales.	mié 01/02/23	jue 02/02/23
8	Reunion de revision del Sprint 1	vie 03/02/23	vie 03/02/23
9	Reunion de planificacion del Sprint 2	sáb 04/02/23	sáb 04/02/23
10	**Realización de pruebas de concepto**	**dom 05/02/2**	**vie 24/02/23**
11	Configuración de un contenedor Docker con dos máquinas Ubuntu.	lun 06/02/23	jue 09/02/23
12	Ejecución de pruebas con las herramientas identificadas para causar denegación de servicio.	vie 10/02/23	jue 23/02/23
13	Selección de la herramienta de ataque para la recopilación de datos de flujo.	vie 24/02/23	vie 24/02/23
14	Reunion de revision del Sprint 2	lun 27/02/23	lun 27/02/23
15	Reunion de planificacion del Sprint 3	mar 28/02/2	mar 28/02/2
16	**Estudio e Implementación de Dorothea**	**mié 01/03/2**	**mar 21/03/2**
17	Investigación sobre la herramienta Dorothea	mié 01/03/2	vie 03/03/23
18	Implementación de la herramienta de denegación de servicio	lun 06/03/23	jue 09/03/23
19	Creación de scripts en Python para iniciar y finalizar el ataque.	vie 10/03/23	mié 15/03/23
20	Pruebas para validar el ataque en Dorothea.	jue 16/03/23	mar 21/03/2
21	Reunion de revision del Sprint 3	mié 22/03/2	mié 22/03/2
22	Reunion de planificacion del Sprint 4	mié 22/03/2	mié 22/03/2
23	**Recopilación de Datos Benignos y de Denegación**	**vie 24/03/23**	**jue 13/04/23**
24	Ejecución y recolección de datos con un atacante	vie 24/03/23	mar 28/03/2
25	Ejecución y recolección de datos con cinco atacantes	mié 29/03/2	vie 31/03/23
26	Ejecución y recolección de datos con diez atacantes.	lun 03/04/23	mié 05/04/2
27	Ejecución y recolección de datos con muestreo (sampling).	jue 06/04/23	mar 11/04/23
28	Recopilación de datos de flujo benignos.	mié 12/04/2	jue 13/04/23
29	Reunion de revision del Sprint 4	vie 14/04/23	vie 14/04/23
30	Reunion de planificacion del Sprint 5	lun 17/04/23	lun 17/04/23
31	**Tratamiento de los conjuntos de datos**	**mar 18/04/2**	**vie 19/05/23**
32	Investigación Teórica del Aprendizaje Automático	mar 18/04/2	vie 28/04/23
33	Preprocesamiento de los datos recolectados	lun 01/05/23	jue 11/05/23
34	Aplicación de algoritmos de aprendizaje automático.	vie 12/05/23	mar 16/05/23
35	Observación de indicadores de rendimiento	mié 17/05/2	vie 19/05/23
36	Reunion de revision del Sprint 5	lun 22/05/23	lun 22/05/23
37	Reunion de planificacion del Sprint 6	mar 23/05/2	mar 23/05/2
38	**Elaboración de informe técnico**	mar 23/05/2	mié 19/07/2
39	Reunion de revision del Sprint 6	jue 20/07/23	jue 20/07/23

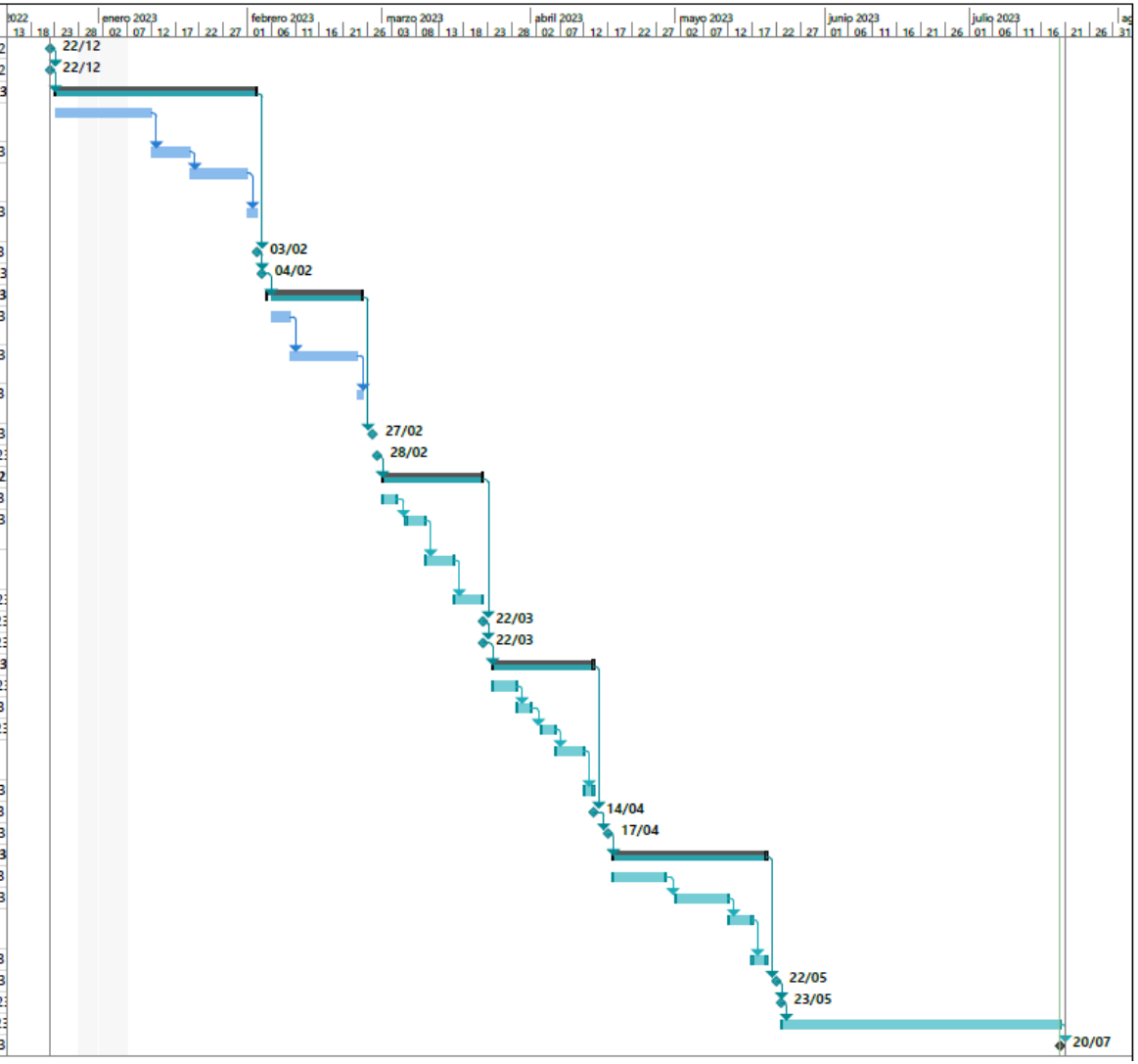

Figura A.1. Diagrama de Gantt

Anexo B: Control de versiones

La herramienta empleada como repositorio de código y control de versiones ha sido GitHub. Como se puede ver en la siguiente URL (https://github.com/fferni01/TFM_Fernando), Dentro del repositorio podemos encontrar 4 carpetas:

- **PoC:** Esta carpeta contiene las distintas pruebas de concepto llevadas a cabo para la selección de la herramienta de denegación de servicio.
- **DOROTHEA-main:** en esta carpeta se encuentra la herramienta Dorothea Utilizada con la herramienta de denegación de servicio implementada.
- **Datos generados:** Contiene los datasets de tráfico generados.
- **Análisis con Machine Learning:** En esta carpeta podemos encontrar los distintos archivos Python que contienen el preprocesamiento de los datasets y los resultados de aplicar aprendizaje automático.
- **Documento-**TFM: Pdf correspondiente Trabajo de fin de máster.