Implementación de un escenario emulado de detección de ataques de ciberseguridad en sistemas de comunicación de nueva generación que involucre la comunicación entre plano de control y plano de datos SDN

Natalia Pérez Puentes

Proyecto de investigación para optar al título de Ingeniera de Telecomunicaciones

Asesor

Ph.D Juan Felipe Botero Vega

Universidad de Antioquia

Facultad de Ingeniería

Ingeniería de Telecomunicaciones

Medellín

2023

Cita	Muñoz Zapata y Martínez Naranjo, 2018 [1]
Referencia Estilo IEEE (2020)	[1] Pérez Puentes, N. "Implementación de un escenario emulado de detección de ataques de ciberseguridad en sistemas de comunicación de nueva generación que involucre la comunicación entre plano de control y plano de datos SDN", proyecto de investigación, Ingeniería de Telecomunicaciones, Universidad de Antioquia, Medellín, Antioquia, 2023.

Grupo de Investigación en Telecomunicaciones Aplicadas (GITA).

UNIVERSIDAD DE ANTIOQUIA
Vicerrectoría de Docencia

Sistema de Bibliotecas

Centro de Documentación Ingeniería (CENDOI)

Repositorio Institucional: http://bibliotecadigital.udea.edu.co

Universidad de Antioquia - www.udea.edu.co

Rector: John Jairo Arboleda Céspedes.
Decano/Director Julio César Saldarriaga Molina.
Jefe departamento: Eduard Emiro Rodríguez Ramírez.

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

Siglas, acrónimos y abreviaturas

P4	Programming Protocol-independent Packet Processors
SDN	Software Defined Networks
DoS	Denial of Service
DDoS	Distributed Denial of Service
MLP	Multilayer Perceptron
RF	Random Forest
SVM	Support Vector Machine
KNN	K-Nearest Neighbor
ONOS	Open Network Operating System

RESUMEN

El escaneo de puertos a menudo es utilizado en la identificación de vulnerabilidades y en la preparación de ataques cibernéticos, lo que resalta la importancia de prevenirlo y detectarlo en los sistemas de comunicación. En el contexto de las redes definidas por software (SDN), que se basan de la separación del plano de control, encargado de tomar decisiones y gestionar las políticas de la red, y el plano de datos, responsable del reenvío de paquetes según las decisiones del plano de control, se han desarrollado sistemas de detección de ataques, enfocados principalmente en ataques de denegación de servicio distribuído (DDoS). Estos sistemas se han implementado tanto en entornos SDN tradicionales como en entornos que incluyen también la programación del plano de datos. Sin embargo, la detección de escaneo de puertos en redes SDN ha recibido poca atención en la literatura, a pesar de que este ataque se emplea comúnmente como una fase inicial en la preparación de otros ataques cibernéticos contra sistemas de comunicaciones.

Este trabajo propone un entorno colaborativo entre el plano de datos y el plano de control para detectar el escaneo de puertos. En este enfoque, el plano de datos proporciona al plano de control la información necesaria para el cálculo de características y su posterior uso en el entrenamiento de modelos de aprendizaje automático. Dichos modelos fueron entrenados con diferentes algoritmos, donde el mejor resultado se obtuvo con el algoritmo Random Forest (RF), con una exactitud superior al 97 %. Este valor supera los resultados obtenidos con este sistema en la detección de ataques DDoS y la forma de calcular las características reduce la sobrecarga del plano de control.

Palabras clave **— SDN, plano de datos, plano de control, escaneo de puertos, aprendizaje automático, clasificación**

I. INTRODUCCIÓN

La implementación de tecnologías SDN ha revolucionado la forma en que se gestionan las redes de comunicación al ofrecer una mayor flexibilidad y un control centralizado mediante la separación de dos componentes esenciales: el plano de datos y del plano de control. En las redes tradicionales, estos planos coexisten en un mismo dispositivo, lo que implica que un solo equipo de red se encarga tanto de transmitir los datos (plano de datos) como de tomar decisiones sobre su enrutamiento y gestión (plano de control). Esta separación ha demostrado ser un avance crucial, ya que en las redes tradicionales, la falta de flexibilidad dificulta la identificación de amenazas en la red, lo que, a su vez, permite a los atacantes evadir las medidas de seguridad con mayor facilidad [2].

En los últimos años, se ha investigado sobre la colaboración del plano de datos programable y el plano de control para la detección de ataques en redes de comunicación [3]. Para lograr esta colaboración, es necesario analizar y procesar una gran cantidad de información relacionada con el estado de la red, por lo que delegar esta tarea únicamente al plano de control puede generar sobrecargas y limitar el rendimiento. La inclusión de la programación del plano de datos ofrece una solución viable al permitir la extracción de parámetros de tráfico para la detección de ciberataques sin afectar el rendimiento del plano de control. Aunque la programación del plano de datos tiene limitaciones en cuanto a capacidad y resolución de cálculos, esta sigue siendo una de las mejores opciones para mejorar la seguridad de las redes de comunicación [4].

Por otra parte, antes de que una red o sistema sea atacado, generalmente se realiza un escaneo de puertos para descubrir vulnerabilidades y posteriormente llevar a cabo otro tipo de ataques más letales [5]. Este escaneo se considera una técnica de reconocimiento que permite al atacante recopilar información, convirtiéndolo en uno de los ciberataques más frecuentes. De acuerdo con CERT [6], un centro brasileño que monitorea informes de incidentes cibernéticos, durante el periodo de enero a agosto de 2023 se presentaron 307.723 incidentes de escaneo de puertos, lo que representa aproximadamente el 70 % de todos los incidentes cibernéticos reportados en Brasil en dicho periodo. Estos datos resaltan la urgente

necesidad de reforzar las defensas en los sistemas de comunicaciones, donde una de las posibles soluciones es la implementación de sistemas que permitan detectar oportunamente este tipo de actividad maliciosa.

II. OBJETIVOS

1. Objetivo general

Implementar un escenario emulado de detección de ataques de ciberseguridad en sistemas de comunicación de nueva generación que involucre la comunicación entre plano de control y plano de datos SDN.

2. Objetivos específicos

- Realizar una revisión de las implementaciones que conectan planos de control con planos de datos programables y elegir una de ellas para reproducirla en el escenario de pruebas.
- Elegir un ataque de ciberseguridad en redes de comunicaciones y describir los parámetros que se deben obtener desde el plano de datos para detectar estos ataques.
- Implementar un plano de datos programable que extraiga los parámetros anteriormente definidos del tráfico y los envíe al plano de control para la posterior detección del ciberataque.
- Implementar una aplicación en el plano de control que usando los parámetros provistos por el plano de datos detecte la existencia del ciberataque elegido a la red de comunicaciones.
- Evaluar la implementación realizada en un escenario de pruebas controlado donde se genere y detecte el ciberataque elegido.

III. MARCO TEÓRICO

A continuación, se presentan las definiciones de los conceptos más relevantes utilizados en este trabajo. Inicialmente se proporciona una descripción de las Redes Definidas por Software (SDN), seguida por una explicación del tipo de ataque que se busca detectar en este trabajo. Finalmente, se presenta una descripción de las técnicas de aprendizaje automático utilizadas en el sistema desarrollado.

1. Redes Definidas por software

El concepto de Redes Definidas por Software (SDN) hace referencia a una arquitectura emergente que tuvo sus inicios en el año 2008. Esta arquitectura propone principalmente la separación entre el plano de datos y el plano de control. En el contexto de SDN, el plano de datos se refiere a la parte de la red encargada de transmitir los datos de manera efectiva, mientras que el plano de control abarca la parte encargada de tomar decisiones sobre cómo enrutar y gestionar esos datos. Esta separación permite simplificar la gestión de la red al brindar mayor flexibilidad y un control centralizado. Su arquitectura consta de lo siguiente:

- Un controlador, esencial en la arquitectura SDN, constituye una parte fundamental del plano de control. El controlador permite la gestión y el control centralizados, la automatización, mantenimiento y la aplicación de políticas en entornos de red físicos y virtuales.
- Los dispositivos de reenvío son elementos del plano de datos que realizan operaciones básicas como el reenvío de paquetes y cuentan con un conjunto de instrucciones claramente definido.
- La interfaz southbound la cual permite la transmisión de información entre el controlador y los dispositivos de red individuales (como conmutadores, puntos de acceso, enrutadores y firewalls).
- La interfaz northbound, la cual conecta las aplicaciones y servicios con el controlador SDN. Normalmente, esta interfaz abstrae los conjuntos de instrucciones de bajo nivel utilizados por las interfaces southbound para la programación de dispositivos de reenvío.

- El plano de gestión, conformado por el conjunto de aplicaciones que aprovechan las funciones ofrecidas por la interfaz northbound para implementar la lógica de control y operación de la red, tales como enrutamiento, firewalls, balanceadores de carga, etc [7].

La Figura 1 ilustra la arquitectura de SDN. Estas redes programables se han aplicado en diferentes contextos, incluyendo redes de área amplia, centros de datos, redes inalámbricas de sensores e incluso, en redes Smart Grid, entre otros dominios. A continuación, se proporciona una descripción más detallada de los planos de control y de datos.

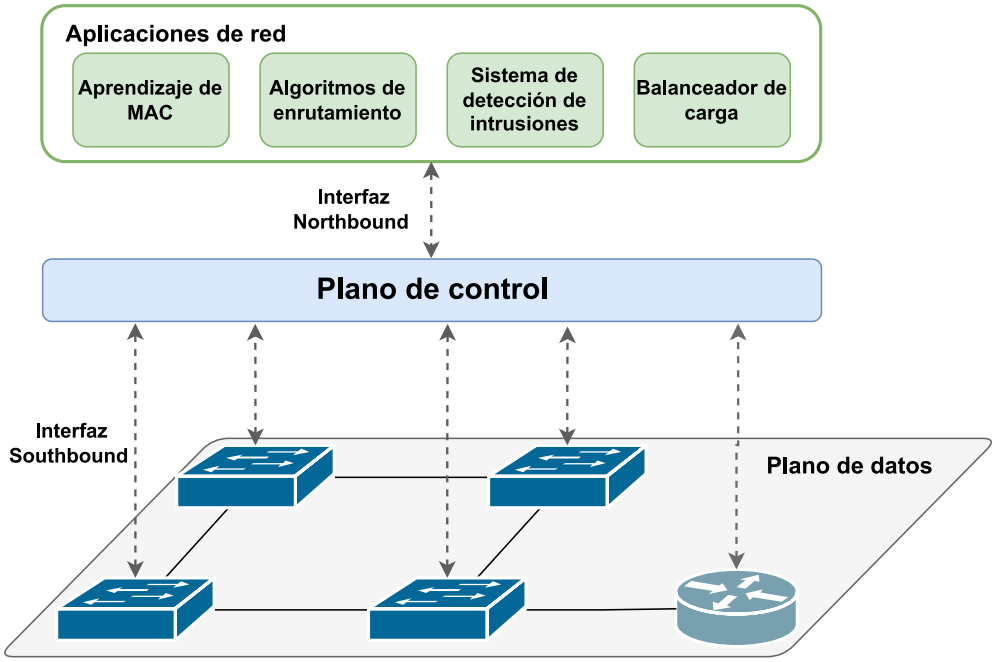

Fig. 1. Arquitectura SDN

1.1. Plano de control:

SDN fue el primer intento de hacer que los dispositivos de red y específicamente su plano de control, fueran programables. En algunos dispositivos seleccionados, los fabricantes permitieron a los usuarios eludir los algoritmos integrados en el plano de control e introducir los suyos propios [3].

El plano de control se encarga de gestionar los dispositivos de reenvío mediante el uso de información y conocimientos de la red global para la toma de decisiones. El controlador SDN, por su parte, traduce los requisitos de las aplicaciones de red que se ejecutan en este en reglas de flujo de bajo nivel, que comparte con los dispositivos SDN a través de la interfaz southbound. Entre las funcionalidades esenciales que ofrece el controlador se encuentran la gestión de topología, la gestión de flujos, la recopilación de estadísticas de los dispositivos SDN y la administración de dispositivos para descubrir tanto los dispositivos de red como los de usuario final que conforman la infraestructura de la red [8].

1.2. Plano de datos:

El plano de datos en una red está compuesto por una serie de elementos de red que se encargan de reenviar paquetes siguiendo reglas de flujo específicas. La comunicación entre el plano de control y el plano de datos se establece a través de la interfaz abierta hacia el sur (conocida como interfaz southbound), la cual es independiente del proveedor de la red. Los algoritmos en el plano de datos tienen la tarea de procesar todos los paquetes que circulan por un sistema de telecomunicaciones, y en gran medida, estos algoritmos definen la funcionalidad, el rendimiento y la escalabilidad de dichos sistemas.

Aunque con limitaciones en cuanto a recursos y capacidades físicas, los planos de datos programables permiten la extracción eficiente de información detallada sobre el tráfico y brindan a los usuarios la flexibilidad para implementar sus propios algoritmos en dispositivos de reenvío. Esto incluye la capacidad de definir nuevos encabezados de protocolo o especificar el comportamiento de reenvío deseado, entre otras.

1.2.1. P4 (Programming Protocol-independent Packet Processors): P4 [9] se ha consolidado como el estándar de facto en la programación del plano de datos en redes. Esta idea y su nombre original se introdujeron en 2014, y desde entonces, ha sido objeto de desarrollo y estandarización por parte del P4 Language Consortium. P4 permite definir y manipular metadatos de paquetes, como “timestamps”, características, estados, latencia, entre otros. Además, P4 permite el uso de medidores, contadores y registros, a los que se puede

acceder tanto para lectura como para escritura y utilizar dentro del programa [10].

Para utilizar P4, el código debe ser compilado y, posteriormente, ejecutado en un dispositivo que cuente con hardware de red programable. En la Figura 2, se ilustra el flujo de trabajo de un programa P4, el cual está estructurado por los siguientes componentes principales [10], [11]:

- **Parser (Analizador):** este componente se encarga del análisis de los paquetes de entrada y la identificación de la pila de protocolos que contienen. Su principal tarea es descomponer los paquetes de entrada en sus diversos campos o cabeceras. Esto es esencial para manipular los datos contenidos en el paquete y procesarlo posteriormente.
- **Tablas de acción:** las tablas son fundamentales en P4 y contienen las reglas de procesamiento de paquetes en forma de "match-action" (coincidencia-acción). Durante el procesamiento de paquetes, el programa P4 busca la regla que coincida con el paquete entrante y aplica la acción asociada. Las reglas pueden abarcar diversas acciones, como actualizaciones de campos de protocolo o selección de puertos, entre otras.
- **Bloques de control (ingreso/egreso):** estos bloques se utilizan para definir como se procesan los paquetes de red. En estos, se determina el orden de las tablas de acción que se aplican a un paquete. Además, ofrecen la posibilidad de implementar condicionales (if, else), utilizar operadores lógicos, realizar operaciones binarias básicas (como suma y multiplicación) y manipular cadenas de bits. Es importante destacar que en los bloques de control de P4 no se permiten ciclos ni operaciones de división.
- **Deparser**: este componente realiza el proceso opuesto al componente 'Parser', es decir, se encarga de reensamblar las cabeceras, así como de añadir nuevas o eliminar las existentes.

2. Escaneo de puertos (port scanning)

El objetivo del escaneo de puertos es identificar las áreas vulnerables de un recurso objetivo a través de las cuales podría ser posible su explotación. Durante este proceso, con el uso de las herramientas de escaneo de puertos se obtiene información valiosa, como la topología de la red, la ubicación de centros de datos, el sistema operativo y la cantidad de

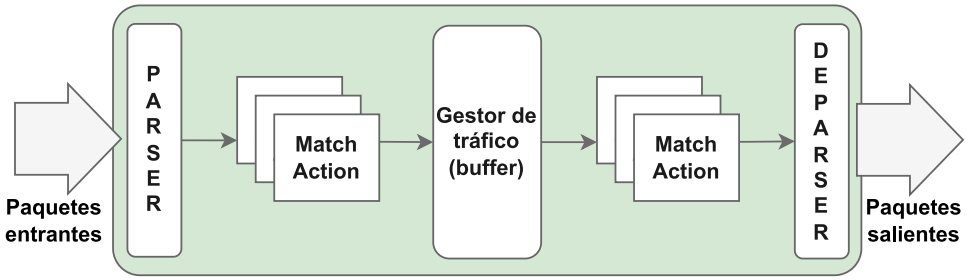

Fig. 2. Flujo de trabajo del plano de datos programable P4

memoria ocupada, entre otros datos. Esto es posible ya que cada servicio o aplicación en un servidor o dispositivo se comunica a través de un puerto específico. Esta información puede ser utilizada para planificar y llevar a cabo una posterior explotación de dicho recurso [12].

Es importante tener en cuenta que el escaneo de puertos no constituye un ataque en sí mismo, sino que forma parte de la fase de reconocimiento que precede a otros tipos de ataques. No obstante, esto no disminuye su relevancia, ya que el conocimiento adquirido durante el escaneo de puertos puede ser utilizado en futuros ataques cibernéticos.

En los últimos años, para la detección de ataques en redes de comunicaciones se suele incorporar en los sistemas de detección desarrollados una o varias técnicas de aprendizaje automático.

3. Algoritmos de aprendizaje automático

El campo del aprendizaje automático se dedica al desarrollo de sistemas que pueden aprender automáticamente de los datos e identificar patrones. Las técnicas de aprendizaje automático se suelen clasificar en tres categorías principales:

- **Aprendizaje supervisado:** En este enfoque, los algoritmos aprenden a partir de datos de entrada y de salida etiquetados durante la fase de entrenamiento. Esto les permite predecir o clasificar nuevos datos basándose en patrones aprendidos previamente.
- **Aprendizaje no supervisado:** En este caso, los modelos se entrenan utilizando datos sin etiquetas. El objetivo es encontrar estructuras o patrones ocultos en los datos, lo que

facilita la identificación de relaciones y agrupaciones sin información previa de estas.

- **Aprendizaje semisupervisado:** En este enfoque, los algoritmos aprovechan tanto datos etiquetados como datos sin etiquetar. Esta combinación permite un aprendizaje más completo y puede mejorar la precisión de las predicciones.

En este trabajo, todos los modelos que se utilizan pertenecen al aprendizaje supervisado, entre los cuales se incluyen:

3.1. Perceptron Multicapa (MLP)

El perceptrón multicapa, MLP por sus siglas en inglés, es un tipo de red neuronal artificial ampliamente utilizada en el aprendizaje supervisado en tareas de clasificación o regresión. La estructura del MLP consta de múltiples capas de neuronas interconectadas, donde las salidas de las neuronas de una capa sirven como entradas para la siguiente capa. Estas capas incluyen una capa de entrada, una capa de salida y una o más capas ocultas, como se muestra en la Figura 3. Cada neurona en las capas ocultas y de salida realiza operaciones matemáticas y aprende a representar características o patrones de los datos mediante el uso de pesos. Cada neurona se modela matemáticamente basándose en el funcionamiento de las neuronas biológicas.

Durante el proceso de entrenamiento, los pesos y el sesgo (bias) se ajustan con el objetivo de minimizar el error en las predicciones. Este algoritmo se emplea comúnmente en problemas no lineales y se basa en la técnica de propagación hacia atrás, también conocida como *backpropagation*, que ajusta los pesos de las conexiones de la red con el objetivo de minimizar el error en la salida [13].

Dentro de los parámetros que se pueden ajustar en el algoritmo MLP, se incluyen el número de capas ocultas, la tasa de aprendizaje, el número máximo de iteraciones, la función de activación y el uso del *early stopping* para la terminación anticipada del entrenamiento en caso de que no se observe una mejora en la puntuación.

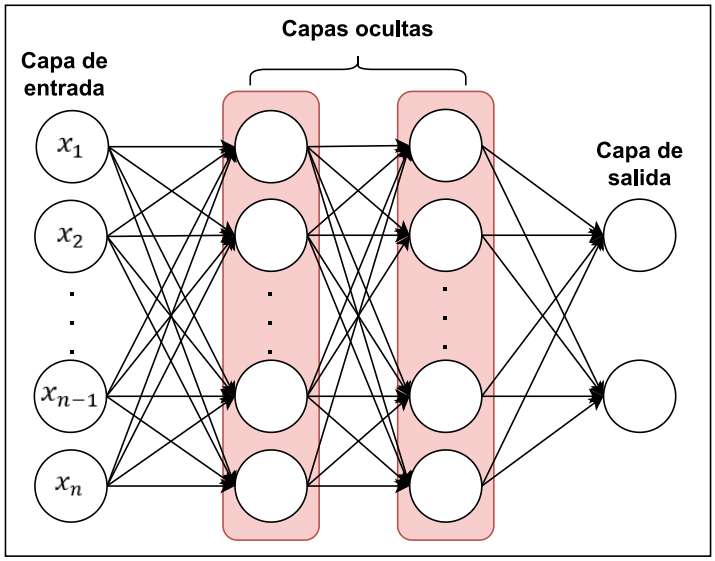

Fig. 3. Clasificador MLP

3.2. Bosque Aleatorio (RF)

Random Forest o bosques aleatorios son un algoritmo de aprendizaje automático supervisado utilizado ampliamente en problemas de regresión y clasificación. Los bosques aleatorios son muy utilizados en diferentes problemas dada su capacidad para proporcionar resultados sólidos incluso sin ajustes exhaustivos de sus hiperparámetros.

Este algoritmo se basa en la combinación de los resultados de múltiples árboles de decisión para llegar a una predicción única. Un árbol de decisión es una estructura jerárquica utilizada para tomar decisiones basadas en datos de entrada. En estos árboles, las características del conjunto de datos se utilizan para dividir los datos en subconjuntos más pequeños siguiendo reglas lógicas. Cada subdivisión conduce a una rama del árbol y, finalmente, a una hoja que representa una decisión o predicción final.

Los bosques aleatorios, en lugar de confiar en un solo árbol, construyen una colección o "bosque" de árboles. Cada árbol en el bosque se entrena de forma independiente con un conjunto de datos de entrenamiento diferente, gracias al muestreo aleatorio (bootstrap) y la selección aleatoria de características. Esta diversidad entre los árboles del bosque hace que

el algoritmo sea altamente resistente al sobreajuste, es decir, es menos propenso a adaptarse demasiado a los datos de entrenamiento y, en cambio, generaliza mejor a nuevos datos.

En problemas de clasificación, después de construir el conjunto de árboles de decisión, se utiliza una técnica conocida como el "voto mayoritario" para tomar decisiones finales [14]. La Figura 4 ilustra el proceso que lleva a cabo este algoritmo para clasificar una nueva instancia.

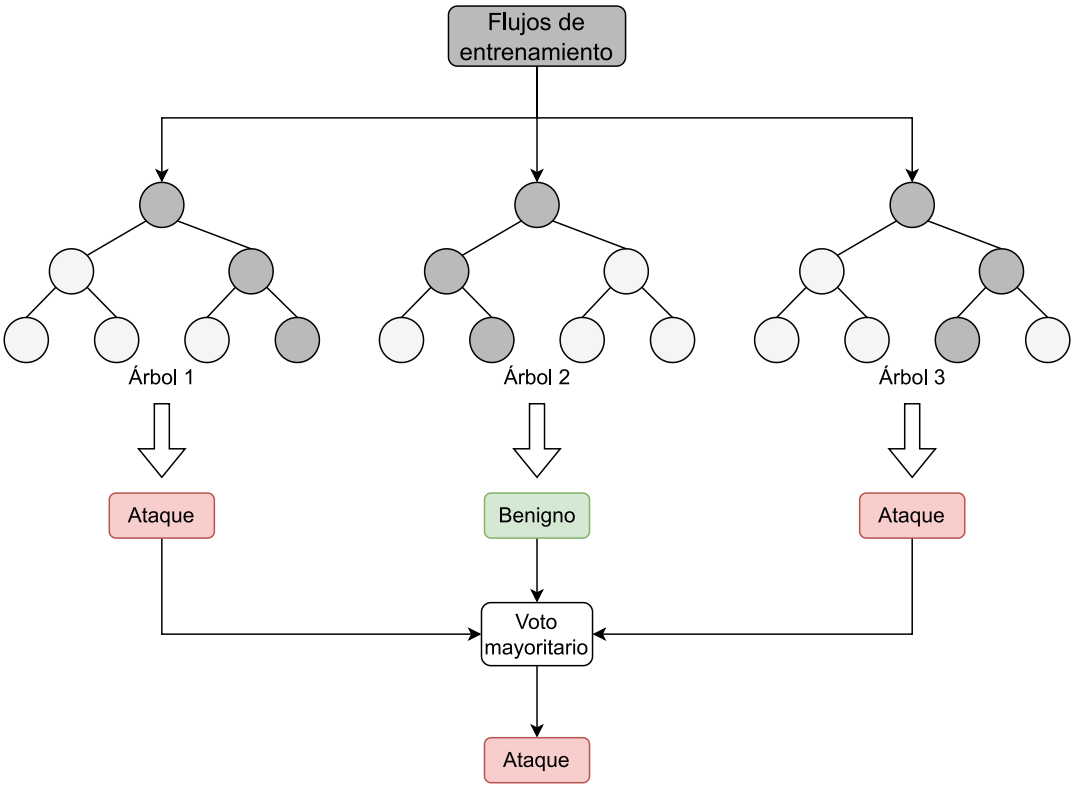

Fig. 4. Clasificador RF

3.3. K-vecinos más cercanos (KNN)

K-vecinos más cercanos, comúnmente conocido como KNN por sus siglas en inglés, es un algoritmo de aprendizaje supervisado basado en instancias que se utiliza tanto en problemas de clasificación como de regresión. A diferencia de otros algoritmos que crean modelos predictivos explícitos, KNN memoriza los datos de entrenamiento y clasifica nuevos

datos teniendo en cuenta la distancia entre ellos y los 'k' datos de entrenamiento más cercanos. En otras palabras, clasifica un punto de datos en función de cómo están clasificados sus vecinos más cercanos.

Es importante destacar que el valor de 'k' no puede ser mayor que la cantidad de datos con los que se entrena el algoritmo, y además, debe ser un número impar mayor que uno [15]. La proximidad o distancia entre los datos y sus vecinos se calcula utilizando funciones como la distancia euclidiana, la similitud coseno, la distancia de Manhattan o la distancia de Minkowski, entre otras. En la Figura 5, se ilustra la influencia del parámetro 'k' cómo el parámetro 'k' afecta la clasificación de una nueva instancia en un escenario de clasificación de tráfico en redes de comunicaciones. Cuando el valor de 'k' es igual a 3, la nueva instancia se clasifica como tráfico benigno, mientras que cuando 'k' aumenta a 5, la nueva instancia se clasifica como ataque.

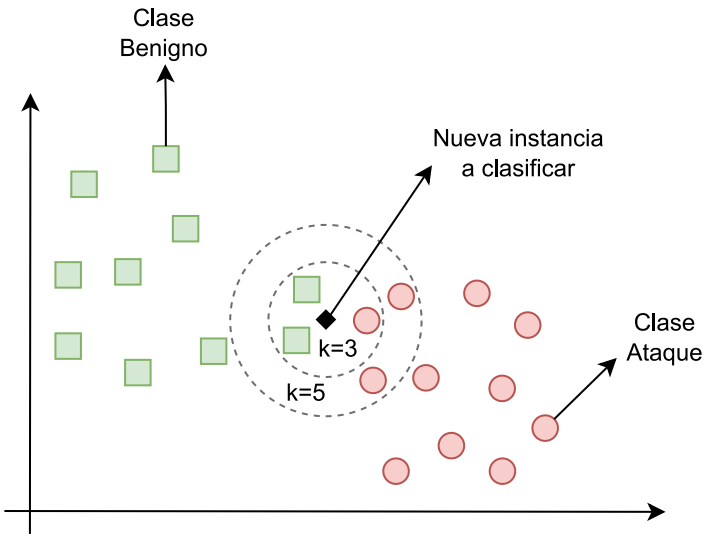

Fig. 5. Lógica de KNN

3.4. Máquinas de soporte vectorial (SVM)

Las Máquinas de Soporte Vectorial (SVM, por sus siglas en inglés) son algoritmos de aprendizaje automático supervisado ampliamente utilizados en problemas de clasificación.

Las SVM se basan en la idea de encontrar un hiperplano en un espacio de N dimensiones (donde N depende de la cantidad de características que tenga el problema de clasificación) que logre la mejor separación posible entre las clases del conjunto de datos.

Los hiperplanos se denominan límites de decisión, y cuanto más alejados estén los datos de este límite, mayor será la confianza de que están bien clasificados. Aunque existen múltiples hiperplanos que pueden separar las clases, el objetivo es maximizar la distancia entre el hiperplano y los puntos de datos más cercanos a él dentro del conjunto de entrenamiento. A esta distancia se le denomina "margen".

Por otra parte, los puntos de datos que se encuentran más próximos al hiperplano y que pueden influir en su la posición y orientación se denominan vectores de soporte. Estos vectores de soporte desempeñan un papel crítico en el proceso de clasificación, ya que se utilizan para maximizar el margen del clasificador. La importancia de estos vectores de soporte es tan significativa que su eliminación podría alterar por completo la posición del hiperplano [16]. La Figura 6, muestra un ejemplo de clasificación en un espacio de dos dimensiones utilizando las SVM.

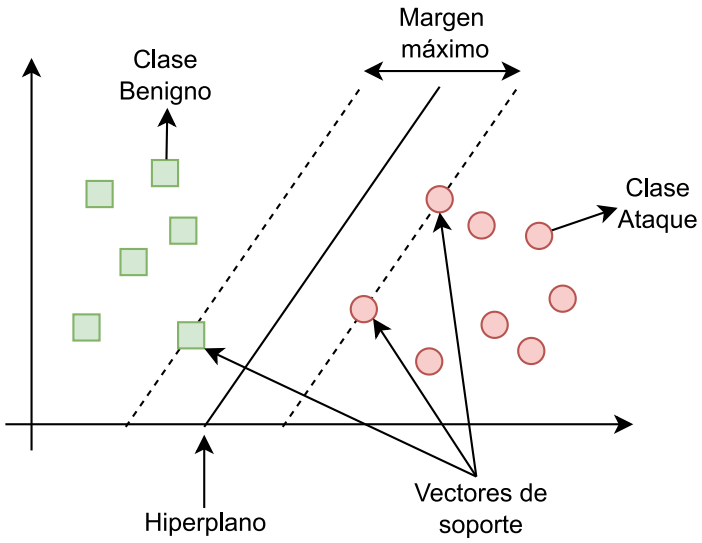

Fig. 6. Lógica de SVM

Entre los parámetros que se pueden ajustar en las SVM se encuentra el parámetro de

regularización y el tipo de kernel utilizado por el algoritmo, entre otros.

IV. TRABAJO RELACIONADO

A continuación, se presentan las principales propuestas para la detección de ataques mediante el uso de SDN, en algunos casos complementadas con modelos de aprendizaje automático. Es importante destacar que la mayoría de estos enfoques se centran en la detección de ataques de denegación de servicio (DoS) en lugar de escaneo de puertos, que es el ataque escogido en este trabajo.

En [17], se propone una estrategia de detección y mitigación de ataques DoS basados en suplantación de identidad en redes SDN. Este enfoque opera en el plano de datos utilizando el lenguaje p4, lo que reduce la carga del controlador. Por su parte en [18], se utiliza el análisis estadístico y teórico para la detección y mitigación de ataques DDoS.

En [19], se introduce P4DDoS, una estrategia de detección de ataques de denegación de servicio distribuido (DDoS) que se basa en las variaciones de la entropía de tráfico de red normalizado. En esta estrategia tanto la estimación del número de flujos distintos en la red como la evaluación de la entropía del tráfico de red normalizado, son implementadas en su totalidad en el plano de datos utilizando P4. Los resultados obtenidos demuestran que esta estrategia logra una precisión de detección comparable a las soluciones más avanzadas disponibles en la actualidad. Además, se distingue por su simplicidad y por la capacidad de ejecutarse íntegramente en el plano de datos.

En el ámbito de la detección mediante técnicas de aprendizaje automático, en [10] se utilizan algoritmos de aprendizaje automático para la detección automatizada de ataques DDoS, en particular, ataques de inundación SYN. Vale la pena mencionar que varios de los autores de este estudio han investigado ampliamente en este campo, abordando preocupaciones relacionadas con la vulnerabilidad de los controladores SDN. Su enfoque busca, en parte, reducir la latencia en la transmisión de datos, y para lograrlo, realizaron la extracción de características en el plano de datos. En este trabajo, se implementaron clasificadores utilizando varios algoritmos, como el RF, KNN, SVM y redes neuronales artificiales (ANN).

Por otro lado, [20] propone DataPlane-ML, una solución que hace uso de modelos de aprendizaje automático y opera en el plano de datos para la detección de ataques de

inundación SYN. En su evaluación, se emplearon los algoritmos KNN, SVM y RF para identificar estos ataques en trazas de red reales. El enfoque de esta propuesta era analizar cómo los diferentes métodos de detección se ven afectados por parámetros como el tiempo de observación y el volumen de paquetes.

Por su parte, [21] propone FlowLens, un enfoque para clasificación en tiempo real de paquetes maliciosos utilizando SDN y aprendizaje automático. Esta solución logra reducir y optimizar el uso de la memoria y tiempo de procesamiento. Los algoritmos de aprendizaje automático que se utilizan en este trabajo son potenciación del gradiente (gradient boosting), RF y Naive Bayes Multinomial.

En [1], se propone ORACLE, una colaboración entre plano de datos y plano de control para detección de ataques DDoS. Con este enfoque se logra extraer características que no son factibles con un enfoque de plano de control que utiliza Open-Flow. En la siguiente sección de este trabajo se profundiza más sobre esta solución.

En cuanto a la detección de escaneo de puertos, en [5] se presenta una solución para prevenir ataques de escaneo de puertos en entornos OpenFlow [22]. En este enfoque, se utilizan estadísticas de flujo recopiladas a intervalos regulares para detectar la presencia de escaneo de puertos. Es importante tener en cuenta que, debido a la recopilación periódica de datos, puede existir un retraso entre la ejecución del escaneo de puertos y su detección. Además, es necesario considerar que a medida que aumenta el número de conmutadores OpenFlow en la red, la carga en el controlador también aumenta, ya que debe recopilar datos estadísticos de todos los conmutadores.

A su vez, en [23] se propone un método de detección de escaneo de puertos basado en las características de los mensajes Packet-In y estadísticas de tráfico en una red Open-Flow. Sin embargo, es importante resaltar que este método no fue probado con tráfico real.

En resumen, estas propuestas abordan la detección de ataques en SDN desde diferentes enfoques. Cada enfoque presenta sus ventajas y desafíos particulares, pero en todos se destaca la importancia de mejorar la seguridad en las redes definidas por software. Es importante destacar que, durante la revisión del estado del arte, se observó que son escasas las estrategias que han desarrollado colaboraciones entre el plano de datos y el plano de control para detectar

ataques de escaneo de puertos. Aunque el escaneo de puertos no se considera un ataque en sí mismo, puede proporcionar al atacante información valiosa para su explotación. Por lo tanto, en este trabajo se propone la detección de escaneo de puertos mediante una colaboración entre el plano de datos y el plano de control, la cual se describe detalladamente en la siguiente sección.

V. SOLUCIÓN PROPUESTA

Este trabajo se enfoca en abordar el desafío de la detección de escaneo de puertos en entornos SDN y se basa en la solución presentada en [1], conocida como ORACLE, una solución que integra el plano de datos y el plano de control para la detección de ataques DDoS en entornos SDN. En la implementación de ORACLE, se utiliza el lenguaje P4 en su versión $P4_{16}$ para la programación del plano de datos y ONOS (Open Network Operating System [24]) como controlador.

ORACLE surgió como una alternativa que aborda dos limitaciones principales en los sistemas de detección de ataques en redes SDN. La primera, se refiere a que la información proporcionada por los contadores de OpenFlow y la que se puede extraer de los paquetes de entrada solo permite calcular algunas de las numerosas características propuestas en la literatura. En segundo lugar, se aborda la sobrecarga del plano de control, que puede surgir debido a grandes volúmenes de tráfico que deben ser considerados en la extracción y clasificación de características. Esta sobrecarga podría aumentar significativamente la latencia y afectar el procesamiento de solicitudes de los dispositivos de la red.

A continuación, se explica cada una de las partes fundamentales para el funcionamiento de ORACLE, que posteriormente se utilizará para la detección de escaneo de puertos.

1. Descripción de ORACLE

1.1. Implementación en el plano de datos utilizando P4

La implementación en el plano de datos se enfoca en la extracción de estadísticas por flujo y su posterior reporte al plano de control. Un flujo se refiere a una secuencia de paquetes de datos que comparten características comunes, como direcciones IP de origen y destino, números de puerto, protocolos de comunicación y otros atributos que permiten agrupar los paquetes relacionados. La comunicación con el plano de control se logra a través de la interfaz P4Runtime, que es una interfaz de tipo southbound diseñada para controlar los elementos del plano de datos que están definidos o descritos por un programa P4 [25].

Para llevar a cabo este proceso, se utilizan ventanas de tiempo. En cada ventana de tiempo, el plano de datos recopila información sobre los flujos que ingresan durante ese período y comunica al plano de control los datos recopilados y procesados durante la ventana de tiempo previa. Para lograr esta funcionalidad, se ha desarrollado una aplicación P4 que consta de tres componentes, los cuales se detallan en Figura 7 y se explican a continuación:

- **Recolector de características:** este componente se encarga de extraer y procesar la información cuando un paquete ingresa al dispositivo. Con esta información, se actualizan las estadísticas de flujo al que pertenece este paquete.
- **Componente de control:** es el encargado de controlar la duración de la ventana de tiempo y verificar en la estructura de datos si existen flujos pendientes de reportar de la ventana de tiempo anterior, antes de que el paquete pase por el componente "recolector de características". Si se identifican flujos que deben ser reportados, el paquete entrante se duplica y se reenvía al "Componente Transmisor de información", acompañado de los datos específicos de cada flujo que se enviarán posteriormente al controlador.
- **Componente de reporte:** es el encargado de transformar el paquete clonado en un paquete de reporte, que luego se reenvía al plano de control a través de la interfaz P4Runtime. Para lograrlo, el paquete clonado se modifica, eliminando su carga útil y las cabeceras innecesarias para la comunicación con el controlador. Luego, se agrega un nuevo encabezado personalizado que contiene la información específica del flujo.

1.2. Mecanismo de almacenamiento en el plano de datos

ORACLE utiliza registros P4 para el almacenamiento de estadísticas de flujo. Un registro es una matriz de tamaño fijo que contiene datos accesibles a través de un índice específico. La singularidad de estos índices, asignados en este caso a cada uno de los flujos unidireccionales, se garantiza haciendo uso de una función hash. Dicha función utiliza los cinco campos que constituyen un flujo como entrada: la dirección IP de origen, la dirección IP de destino, el puerto de origen, el puerto de destino y el protocolo de transporte.

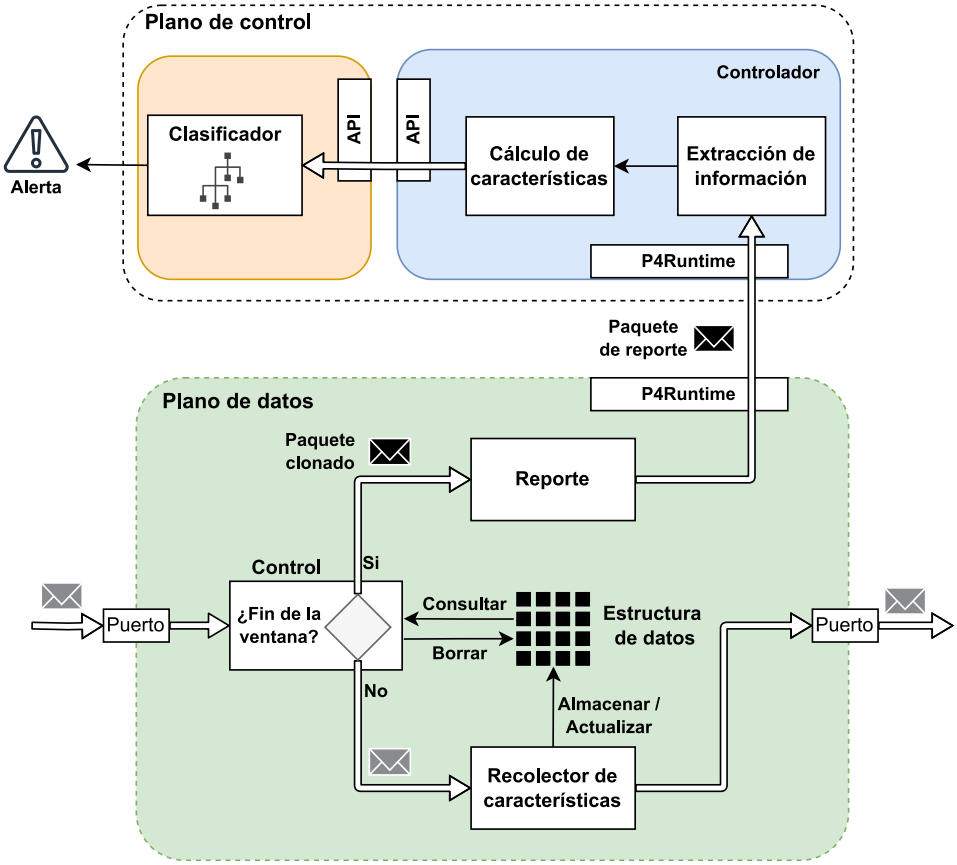

Fig. 7. Arquitectura de ORACLE

Dado que el plano de datos reporta flujos bidireccionales, es necesario contar con los índices para ambas direcciones. Para obtener el índice de un flujo bidireccional, se realiza nuevamente el cálculo de la función hash utilizando una segunda tupla que presenta los mismos atributos, aunque en un orden diferente (ip y puerto origen pasan a ser ip y puerto destino). Los dos índices resultantes de estos cálculos se almacenan en una estructura de datos adicional (ver Figura 8), que se asemeja a un buffer LIFO (Last-In, First-Out). En otras palabras, los índices se almacenan secuencialmente en orden de llegada, y cuando la información debe enviarse al plano de control, se consumen del buffer en el orden opuesto al de llegada hasta que este se encuentre completamente vacío.

Para garantizar que el proceso de recolección de índices no se interrumpa mientras el

buffer se va vaciando, se propone abstraer el concepto de una carretera de dos carriles, cada uno con su propio buffer independiente. Uno de estos carriles se utiliza para la recolección de índices nuevos, mientras que el otro se emplea para informar sobre los flujos almacenados en la ventana de tiempo anterior. Esta estructura de dos carriles permite un flujo continuo de datos sin interrupciones en el proceso de recolección y transmisión de información al plano de control.

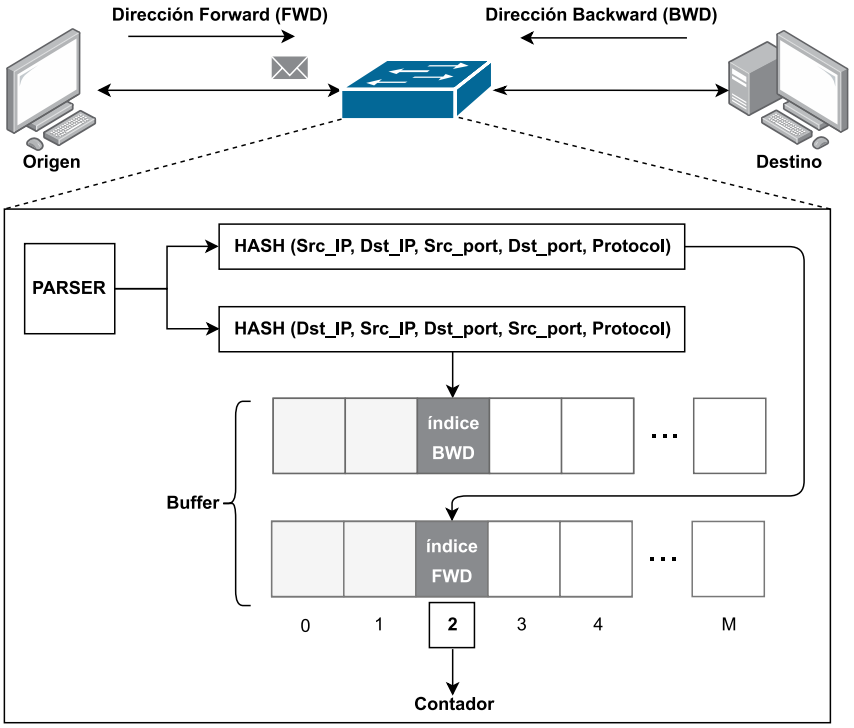

Fig. 8. Mecanismo de almacenamiento de los índices de cada flujo

1.3. Implementación en el plano de control

En este plano, se destacan dos implementaciones. La primera es una aplicación ONOS que se encarga de calcular las características que no se calcularon en el plano de datos. La segunda implementación es una API-REST encargada de la clasificación del tráfico. Para llevar a cabo la clasificación de los flujos reportados, se destacan 3 componentes:

- Extracción de información: En este componente se extrae la información del encabezado personalizado del paquete de reporte y es analizado para obtener información de cada flujo recibido.
- Cálculo de características: Se construye una tupla con las características calculadas a partir de las estadísticas obtenidas en la fase anterior.
- Clasificador: Este componente recibe las características calculadas a través de una petición REST y se encarga de clasificar el tráfico como benigno o ataque DDoS. Si se detecta un posible ataque, se emite una alarma.

Es importante destacar que la API-REST queda a la espera de solicitudes de clasificación solo cuando ya se completó su proceso de entrenamiento, un aspecto que se detallará en una sección posterior.

En la Figura 7 se ilustra la arquitectura de ORACLE con sus implementaciones tanto en el plano de datos como en el plano de control.

1.4. Características utilizadas en la implementación de ORACLE

ORACLE utiliza un conjunto de cuatro características seleccionadas entre 80 posibles, las cuales fueron obtenidas mediante el uso de la herramienta CICFlowMeter [26]. Estas características fueron recomendadas por los autores de la base de datos CICIDS-2017 [27] para llevar a cabo la detección de ataques DDoS y se listan a continuación:

- **Flow Duration (F1):** representa la duración del flujo en microsegundos.
- **Flow-IAT Std (F2):** indica la desviación estándar del tiempo de llegada entre paquetes consecutivos del mismo flujo.
- **avgPacketSize (F3):** corresponde al tamaño promedio de la carga útil de los paquetes de un mismo flujo.
- **Bwd Packet Length Std (F4):** representa la desviación estándar del tamaño de la carga útil de los paquetes del mismo flujo, en dirección de retorno.

Es importante destacar que para el cálculo de F1, F2 y F3 se consideran las estadísticas en ambas direcciones del flujo, mientras que para el cálculo de F4 solo se tienen en cuenta

las estadísticas recolectadas en la dirección de retorno del flujo (bwd). Además, F1 y F3 son características a nivel de flujo, lo que significa que se calculan considerando todos los paquetes del flujo, mientras que F2 y F4 son características intra-flujo, lo que implica que se aplican a paquetes consecutivos dentro del mismo flujo. Las características a nivel de flujo pueden obtenerse en un entorno SDN-OpenFlow mediante el uso de las estadísticas por flujo proporcionadas por esta arquitectura. Sin embargo, las características a nivel intra-flujo representan un desafío adicional, ya que no es posible obtenerlas en dicho entorno. El cálculo de estas características es una de las contribuciones distintivas de ORACLE.

Se debe tener en cuenta que F1 es calculada en el plano de datos, mientras que F2, F3 y F4 son calculadas en el plano de control utilizando las estadísticas proporcionadas por el plano de datos. Esto se debe a las limitaciones del plano de datos para realizar divisiones o cálculos de raíces cuadradas. F2, F3 y F4 se basan en operaciones de promedio y desviación estándar, que se calculan utilizando las ecuaciones 1 y 2. En dichas ecuaciones, x_i es la diferencia de tiempo de llegada entre el paquete actual y el anterior del mismo flujo para F2, mientras que para F3 y F4 x_i es el tamaño en bytes de la carga útil del paquete i de n paquetes pertenecientes al mismo flujo.

$$\overline{x} = \frac{\sum_{i=1}^{n} x_i}{n-1} \tag{1}$$

$$std = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \overline{x})^2}{n-1}} = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - 2\overline{x}\sum_{i=1}^{n} x_i + n\overline{x}^2}{n-1}} \tag{2}$$

Cabe mencionar que la ecuación 2 se descompone utilizando propiedades aritméticas, lo que tiene como resultado una ecuación que depende de dos sumatorias. Dichas sumatorias son calculadas en el plano de datos y reportadas al plano de control, lo que evita un alto consumo de recursos al enviar los resultados de esas sumatorias y no cada valor individual de x_i al plano de control. Con esta propuesta, el plano de control solo debe calcular las características y reportarlas a la API-REST, evitando su sobrecarga.

1.4.1. Información reportada por el plano de datos: para que el plano de control pueda realizar el cálculo de las cuatro características mencionadas anteriormente, se precisa que el plano de datos envíe las estadísticas que se muestran en la Tabla I, donde también se incluye F1.

Característica/Estadística	**Descripción**	**Tamaño (bytes)**
Flow Duration (F1)	Duración del flujo en microsegundos.	6
Tot Fwd Pkt (S2)	Número total de paquetes en la dirección de ida (fwd).	4
Tot Bwd Pkt (S3)	Número total de paquetes en la dirección de retorno (bwd).	4
Tot Len Fwd Pkt (S4)	Sumatoria de tamaños de carga útil de los paquetes en dirección de ida ($\sum x_i$).	4
Tot Len Bwd Pkt (S5)	Sumatoria de tamaños de carga útil de los paquetes en dirección de retorno ($\sum x_i$).	4
Tot Len Bwd Pkt Sqrt (S6)	Sumatoria de tamaños al cuadrado de carga útil de los paquetes en dirección de retorno ($\sum {x_i}^2$).	5
IAT Total (S7)	Sumatoria de tiempos de llegada entre paquetes ($\sum x_i$).	6
IAT Total Sqrt (S8)	Sumatoria de tiempos al cuadrado de llegada entre paquetes ($\sum {x_i}^2$).	7

TABLA I: Información reportada al plano de control por cada flujo (adaptada de [1])

1.4.2. Cálculo de las características: una vez que las estadísticas llegan al controlador, este procede a realizar el cálculo de las características utilizando las ecuaciones que se muestran en la Tabla II, según corresponda.

2. Adaptación de ORACLE para la detección de escaneo de puertos

Como se mencionó al principio de esta sección, el enfoque principal de este trabajo es utilizar ORACLE para la detección de ataques de escaneo de puertos. Es importante destacar

Característica	**n**	**media** ($\overline{x}$)	**std**
FlowIATStd (F2)	$S2 + S3$	$\frac{S7}{n-1}$	$\sqrt{\frac{S8 - 2 \cdot \overline{x} \cdot S7 + n \cdot \overline{x}^2}{n-1}}$
AvgPacketSize (F3)	$S2 + S3$	$\frac{S4 + S5}{n-1}$	-
BwdPktLenStd (F4)	$S3$	$\frac{S5}{n-1}$	$\sqrt{\frac{S6 - 2 \cdot \overline{x} \cdot S5 + n \cdot \overline{x}^2}{n-1}}$

TABLA II: Ecuaciones para el cálculo de F2, F3 y F4

que las características que ORACLE utiliza en la detección de ataques DDoS son diferentes de las recomendadas por los autores de la base de datos CIC-IDS2017 para la detección de escaneo de puertos. En este contexto, este trabajo propone evaluar el rendimiento de ORACLE en la detección de ataques de escaneo de puertos utilizando las características que normalmente emplea para detectar ataques DDoS.

Dado que ORACLE es un trabajo que comenzó su desarrollo en 2019 y fue publicado en 2021, fue necesario realizar ciertas modificaciones y actualizaciones en algunos de los códigos utilizados en su implementación para garantizar su correcto funcionamiento. Estos cambios se realizaron principalmente en el código P4, ya que algunos métodos han evolucionado desde el inicio del proyecto y en la API-REST para que fuera compatible con Python 3.8, una versión más reciente que la utilizada en su desarrollo original.

Un cambio significativo con respecto al desarrollo original de ORACLE es la adopción del switch Stratum [28] en lugar del switch Bmv2 (behavioral-model version 2) [29]. Stratum es un sistema operativo de conmutador de código abierto para SDN que no está vinculado a interfaces específicas de silicio proporcionadas por un proveedor de hardware específico.

Un aporte significativo de este trabajo es dar al sistema la capacidad para detectar flujos compuestos por un solo paquete como posibles ataques. En la versión original de ORACLE, los flujos solo se etiquetaban como ataques si estaban conformados por dos o más paquetes, lo cual presentaba un desafío particular en el caso del escaneo de puertos, ya que resultaba en la clasificación errónea de muchos ataques como tráfico benigno.

VI. EVALUACIÓN Y RESULTADOS

Para evaluar el rendimiento del sistema en la detección de ataques de escaneo de puertos, se optó por replicar varias trazas de paquetes resultantes del experimento realizado por [27] en un escenario real, las cuales están disponibles para el uso libre de la comunidad científica. El tráfico de ataques de escaneo de puertos se extrajo de un archivo .pcap que también contenía tráfico de los ataques DDoS, LOIT (Low Orbit Ion Cannon) y Botnet. Luego del proceso de filtrado, el tráfico resultante estaba compuesto por 161552 paquetes de escaneo de puertos. Para el tráfico benigno, se extrajo una porción del tráfico de otro archivo .pcap que contenía exclusivamente este tipo de tráfico.

Por otra parte, al igual que en las pruebas originales de ORACLE, se decidió explorar el impacto del sistema al utilizar diferentes ventanas de tiempo en el plano de datos. Esto implica la necesidad de contar con bases de datos separadas para cada ventana de tiempo. Las ventanas de tiempo utilizadas en las pruebas son de 5, 20, 40 y 60 segundos.

A continuación, se describe el escenario experimental, así como la fase de entrenamiento de los modelos utilizados en el sistema de detección.

1. Escenario experimental

El escenario utilizado para llevar a cabo las pruebas en el sistema de detección de escaneo de puertos se detalla en la Figura 9. En este entorno, el plano de datos fue emulado utilizando Mininet y está conformado por un switch Stratum con soporte P4, un host encargado de reproducir el tráfico utilizando la herramienta TCPReplay y un servidor que actúa como la víctima, al cual se reenvía todo el tráfico replicado.

En cuanto al plano de control, este se compone de un controlador ONOS que está conectado al switch Stratum mediante la interfaz P4Runtime. Finalmente, se cuenta con una API-REST donde se cargan los modelos previamente entrenados para la posterior clasificación del tráfico.

Para construir la base de datos necesaria para el entrenamiento de los modelos de aprendizaje automático, ORACLE cuenta con una aplicación ONOS que permite almacenar

las características de cada flujo reportado por el plano de datos con su respectiva etiqueta (tráfico benigno o ataque) en un archivo .csv. Para etiquetar los paquetes, se modificaron los dos últimos bits del campo de cabecera IPV4 “Tipo de Servicio”. Se estableció la convención de utilizar "00"para etiquetar el tráfico como benigno y "11"para etiquetar el tráfico como ataque de escaneo de puertos. Esta modificación permitió obtener las etiquetas una vez el tráfico fue replicado en el plano de datos para posteriormente reportarlas al plano de control.

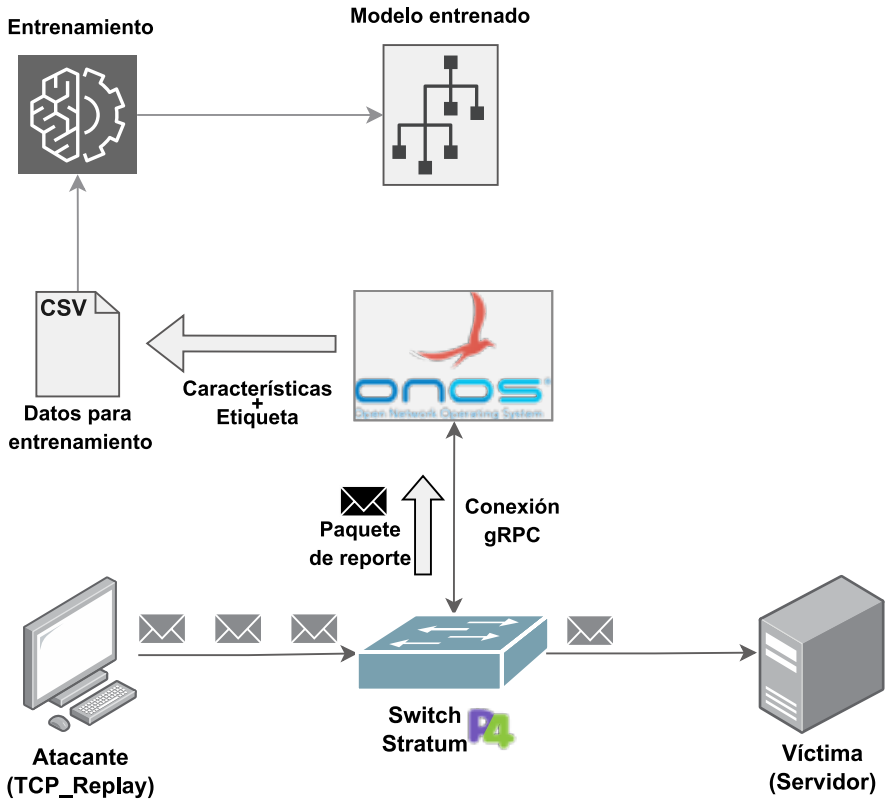

Fig. 9. Configuración experimental del sistema de detección

2. Entrenamiento de los modelos

Después de generar las bases de datos correspondientes a las ventanas de tiempo seleccionadas, se inicia la etapa de entrenamiento de los modelos de aprendizaje automático. En este trabajo se emplean y comparan cuatro clasificadores diferentes por cada ventana de

Ventana de tiempo	Flujos tráfico Portscan	Flujos tráfico Benigno	Total flujos por base de datos
5 segundos	80.181	2.504	82.685
20 segundos	79.027	1.988	81.015
40 segundos	78.148	1.836	79.984
60 segundos	78.429	1.712	80.141

TABLA III: Flujos resultantes luego de replicar el tráfico en el plano de datos

tiempo: MLP, RF, KNN y SVM por lo que en total se evalúan 16 modelos.

Para el proceso de entrenamiento, se optó por utilizar aproximadamente el 60 % de los paquetes disponibles con tráfico de escaneo de puertos, lo que equivale a 96,931 paquetes. Además, se utilizaron 97000 paquetes con tráfico benigno, para un total de 193931 paquetes destinados a la fase de entrenamiento.

Por otra parte, la cantidad de flujos por cada ventana de tiempo después de replicar el tráfico se presenta en la Tabla III. En ella, se observa una reducción en el número de flujos a medida que se incrementa la duración de la ventana. Esta disminución se debe al hecho de que todos los flujos se dirigen al plano de control en la ventana de tiempo siguiente, sin tener en cuenta si han finalizado o no.

Para llevar a cabo el entrenamiento de estos modelos, se aprovechan las bibliotecas de scikit-learn en el lenguaje de programación Python, reconocidas por su eficacia y facilidad de uso en tareas de aprendizaje automático. La Tabla IV detalla los hiperparámetros que se emplearon en cada uno de los modelos.

Para la evaluación de estos clasificadores, se utiliza la matriz de confusión. Una matriz de confusión es una herramienta que permite evaluar el desempeño de algoritmos en el aprendizaje supervisado. Para analizar el desempeño de los modelos en la clasificación de ataques de escaneo de puertos, se obtienen cuatro resultados importantes que posteriormente se utilizan en el cálculo de las métricas de rendimiento:

- **Verdadero positivo (TP)**: indica el número de flujos con tráfico de escaneo de puertos

MLP	RF	KNN	SVM
hidden_layer_sizes: 100 activation: relu max_iter:2000 early_stopping: True	n_estimators: 20	n_neighbors: 3	C: 10 kernel: rfb

TABLA IV: Hiperparámetros de los modelos de aprendizaje automático implementados

clasificados correctamente.

- **Falso positivo (FP):** indica el número de flujos benignos que se clasificaron como flujos de escaneo de puertos.
- **Verdadero negativo (TN):** indica el número de flujos con tráfico benigno clasificados correctamente.
- **Falso negativo (FN):** indica el número de flujos con tráfico de escaneo de puertos que se clasificaron como flujos benignos.

En este caso, las métricas que se utilizan para evaluar el desempeño de los modelos son las siguientes:

- **Precisión:** mide cuántas de las predicciones positivas realizadas por el modelo son correctas en relación con todas las instancias que el modelo ha etiquetado como positivas.

$$Precisión = \frac{TP}{TP + FP} \tag{3}$$

- **Sensibilidad:** también conocida como recall, mide la proporción de instancias positivas que el modelo predijo correctamente en relación con todas las instancias reales que son positivas. Es una medida de cuántos de los casos positivos fueron detectados por el modelo.

$$Sensibilidad = \frac{TP}{TP + FN} \tag{4}$$

- **Exactitud:** mide la cantidad de instancias de datos clasificados correctamente con respecto a la cantidad total de instancias.

$$Exactitud = \frac{TP + TN}{TP + FP + TN + FN} \tag{5}$$

- **F1-score:** es una medida que combina precisión y recall en un solo valor. Es útil cuando se desea encontrar un equilibrio entre la precisión y la capacidad del modelo para capturar todos los casos positivos. Se suele utilizar en problemas donde el conjunto de datos está desbalanceado.

$$F1score = \frac{2 \cdot Precisión \cdot Sensibilidad}{Precisión + Sensibilidad} = \frac{2 \cdot TP}{2 \cdot TP + FN + FP} \tag{6}$$

3. Resultados y análisis

En la fase de prueba del sistema de detección, se utilizó el 40 % restante de los paquetes disponibles que contienen tráfico de escaneo de puertos, lo que equivale a 64.621 paquetes. En cuanto al tráfico benigno, durante la replicación del tráfico para la generación de las bases de datos de entrenamiento, se observó una diferencia significativa en el número de flujos obtenidos por cada tipo de tráfico que se utiliza. Esto se debe a la naturaleza del ataque de escaneo de puertos, que implica el envío de numerosos paquetes SYN a diferentes puertos, generando una gran cantidad de flujos compuestos por un solo paquete. Como resultado, la cantidad de flujos para este tipo de ataque es considerablemente mayor que la del tráfico benigno. Por esta razón, se optó por probar el sistema de detección con 1.000.000 paquetes con tráfico benigno, para evaluar el sistema con un mayor número de instancias desconocidas.

En la Figura 10, se observa la influencia de las ventanas de tiempo en el resultado del F1-score de los diferentes algoritmos implementados en la detección del ataque de escaneo de puertos. En este caso, para las ventanas de 5 y 20 segundos, el algoritmo MLP obtuvo el mejor resultado, mientras que, para la ventana de 40 segundos, el mejor resultado lo obtuvo el RF. En el caso de la ventana de 60 segundos, aunque el MLP mostró un rendimiento similar, las SVM lograron el mejor resultado. Cabe resaltar que, para la ventana de 5 segundos, los algoritmos MLP, KNN Y SVM también obtuvieron resultados muy similares.

Además, se evidencia que para el MLP y las SVM, los resultados no varían significativamente a medida que aumenta el tamaño de la ventana de tiempo, pero se logra apreciar un pico en la ventana de 20 segundos en ambos casos, siendo más pronunciado con el algoritmo MLP. A su vez, en el caso de RF y KNN, se observa una variación significativa que depende

de la ventana utilizada. En todos los casos, el F1-score más bajo se obtuvo con la ventana de 60 segundos.

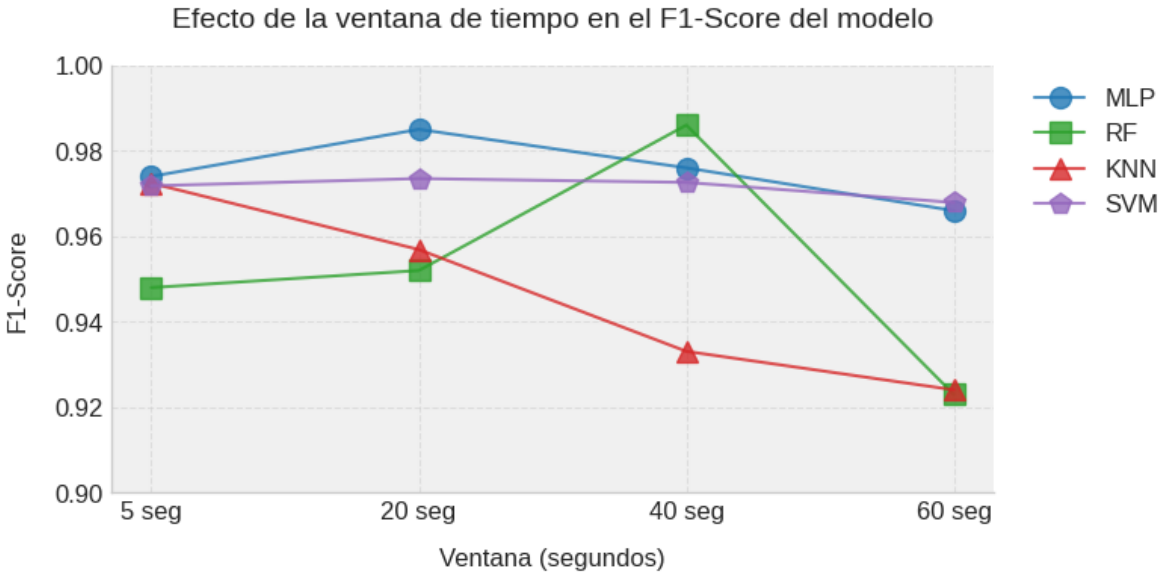

Fig. 10. Resultados del F1-score de todos los modelos para el ataque de escaneo de puertos

La Figura 11, por otro lado, muestra la influencia de la ventana de tiempo en el F1-score de los modelos KNN y RF para el ataque de escaneo de puertos y el ataque DDoS. Los resultados de DDoS se obtienen de [1], de los primeros experimentos de ORACLE. En el caso del ataque de escaneo de puertos, el KNN muestra una disminución en el F1-score a medida que aumenta el tamaño de la ventana, mientras que el RF alcanza un punto máximo con la ventana de 40 segundos, seguido de una disminución con la ventana de 60 segundos. En cuanto al ataque DDoS, tanto KNN como RF muestran un aumento en el F1-score a medida que aumenta la ventana de tiempo, alcanzando el mejor resultado con una ventana de 60 segundos (a diferencia del caso del portscan), aunque estos resultados son muy similares a los obtenidos con la ventana de 40 segundos.

En las Tablas V, VI, VII y VIII se presentan los resultados de todas las métricas para cada ventana de tiempo utilizada en cada uno de los algoritmos de aprendizaje automático. Estas tablas permiten identificar la ventana en la que se obtuvieron los mejores resultados

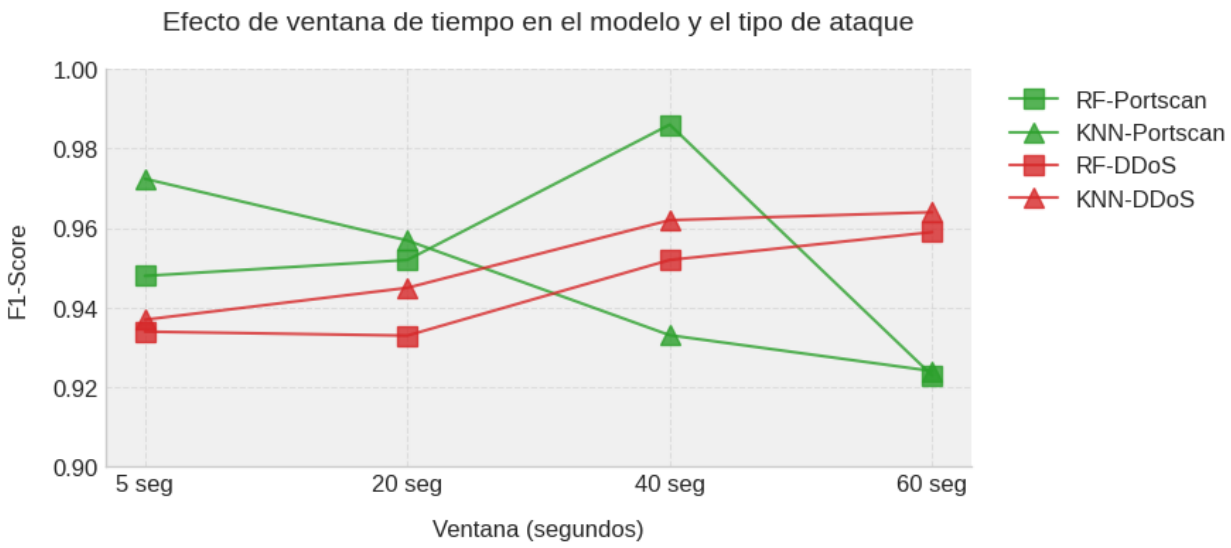

Fig. 11. Resultados del F1-score de los modelos KNN y RF para el ataque de escaneo de puertos y el ataque DDoS

en todas las métricas para cada algoritmo.

En el caso del MLP y SVM, se observa que los resultados no varían significativamente entre las diferentes ventanas de tiempo. Sin embargo, en ambos casos, el mejor resultado se logró al utilizar la ventana de 20 segundos. Cabe destacar que, en el caso de SVM, los resultados entre las ventanas de 5, 20 y 40 segundos fueron muy similares.

En contraste, para el algoritmo RF, el mejor resultado se obtuvo con la ventana de 40 segundos y en el caso del KNN, los mejores resultados se alcanzaron con la ventana de 5 segundos. Finalemnte, el algoritmo que obtuvo el mejor desempeño según las métricas calculadas fue el RF seguido por el MLP, con una diferencia poco significativa.

El "mejor resultado" se refiere al valor más alto de sensibilidad, que indica la capacidad del modelo para detectar ataques de escaneo de puertos, así como al valor más alto de exactitud y de F1-score. Es importante mencionar que, en todos los casos, la precisión fue de aproximadamente el 99 %.

También es relevante señalar que con todos los algoritmos al menos una de las ventanas generó un F1-score superior al 97 %. Esta métrica es particularmente valiosa cuando la base

de datos está altamente desbalanceada, como ocurre en este caso, lo que sugiere una buena capacidad de generalización por parte de los modelos.

Ventana	**Precisión**	**Sensibilidad**	**Exactitud**	**F1-score**
5 seg	0,996	0,953	0,955	0,974
20 seg	**0,991**	**0,980**	**0,974**	**0,985**
40 seg	0,997	0,955	0,958	0,976
60 seg	0,998	0,936	0,94	0,966

TABLA V: Métricas para MLP

Ventana	**Precisión**	**Sensibilidad**	**Exactitud**	**F1-score**
5 seg	0,997	0,903	0,913	0,948
20 seg	0,999	0,91	0,92	0,952
40 seg	**1,000**	**0,974**	**0,976**	**0,986**
60 seg	0,999	0,858	0,867	0,923

TABLA VI: Métricas para RF

Ventana	**Precisión**	**Sensibilidad**	**Exactitud**	**F1-score**
5 seg	**0,996**	**0,950**	**0,954**	**0,972**
20 seg	0,994	0,922	0,927	0,957
40 seg	0,998	0,876	0,887	0,933
60 seg	0,999	0,860	0,871	0,924

TABLA VII: Métricas para KNN

Ventana	**Precisión**	**Sensibilidad**	**Exactitud**	**F1-score**
5 seg	0,996	0,948	0,952	0,972
20 seg	**0,987**	**0,960**	**0,954**	**0,974**
40 seg	0,998	0,949	0,951	0,973
60 seg	0,998	0,940	0,943	0,968

TABLA VIII: Métricas para SVM

VII. CONCLUSIONES

En este trabajo, se propuso detectar ataques de escaneo de puertos en un entorno colaborativo que originalmente se había diseñado para la detección de ataques DDoS llamado ORACLE. Para lograr esta adaptación, fue necesario revisar y ajustar cada uno de los componentes del entorno, incorporando las actualizaciones necesarias y añadiendo funcionalidades específicas para la detección de escaneo de puertos. Este entorno de seguridad aprovecha tanto características a nivel de flujo como a nivel intra-flujo para llevar a cabo la detección. Además, se ha diseñado para reportar los datos recopilados del plano de datos al plano de control en ventanas de tiempo de 5, 20, 40 y 60 segundos.

En el proceso de clasificación, se entrenaron 16 modelos, 8 modelos más que en la propuesta original de ORACLE. Se decidió incluir, además de los algoritmos de aprendizaje automático RF Y KNN, los algoritmos MLP y SVM. En general con todos los modelos probados, se logró un F1-score superior al 92 % en todas las ventanas de tiempo y una exactitud de más de 86 %. Esto es significativo dado que las características utilizadas no son las ideales para la detección de ataques de escaneo de puertos. Además, se lograron entrenar modelos que alcanzaron una exactitud y F1-score de aproximadamente 98 % y 99 %, respectivamente. En este caso, el algoritmo RF demostró ser el más efectivo en la detección de ataques de escaneo de puertos. Vale la pena mencionar que, gracias a las adaptaciones realizadas en el sistema de detección, se lograron métricas más altas que las obtenidas en la implementación original de ORACLE.

En el caso de DDoS, no se consideraron como ataques aquellos flujos compuestos por un solo paquete, mientras que en el contexto del escaneo de puertos, sí se consideraron como ataques. Esta diferencia condujo a una alta cantidad de flujos compuestos por un solo paquete en el caso del escaneo de puertos, fenómeno que puede explicarse por la naturaleza de este tipo de ataque. Como resultado, se tuvieron que entrenar los modelos con una base de datos desbalanceada, ya que en el caso del tráfico benigno, no se generan tantos flujos.

Cabe mencionar que la influencia de las ventanas de tiempo en la detección de ataques de escaneo de puertos no siguió el mismo patrón que en el caso de los ataques DDoS. Para

los ataques DDoS, tanto el algoritmo RF como el KNN obtuvieron los mejores valores de F1-score con la ventana de 60 segundos y los mejores valores de exactitud con la ventana de 40 segundos. En contraste, en el escenario de ataque de escaneo de puertos, cada algoritmo obtuvo sus mejores resultados (los valores más altos de sensibilidad, exactitud y F1-score) en una ventana de tiempo específica, que varió entre los algoritmos utilizados.

Finalmente, como perspectiva para el trabajo futuro, se pretende desarrollar un framework que sea independiente del tipo de ataque, incorporando una gama más amplia de estadísticas para el cálculo de otras características de las propuestas en [27]. Este framework proporcionará la flexibilidad de solicitar y obtener las estadísticas necesarias directamente del plano de datos, permitiendo un enfoque más completo para la detección y mitigación de amenazas en redes SDN.

REFERENCIAS

[1] S. G. Macías, L. P. Gaspary, and J. F. Botero, "Oracle: An architecture for collaboration of data and control planes to detect ddos attacks," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 2021, pp. 962–967.

[2] Y. Gao and Z. Wang, "A review of P4 programmable data planes for network security," *Mobile Information Systems*, vol. 2021, pp. 1–24, 2021.

[3] F. Hauser, M. Häberle, D. Merling, S. Lindner, V. Gurevich, F. Zeiger, R. Frank, and M. Menth, "A survey on data plane programming with p4: Fundamentals, advances, and applied research," *Journal of Network and Computer Applications*, vol. 212, p. 103561, 2023.

[4] B. M. Xavier, R. S. Guimarães, G. Comarela, and M. Martinello, "MAP4: A Pragmatic Framework for In-Network Machine Learning Traffic Classification," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4176–4188, 2022.

[5] C. V. Neu, C. G. Tatsch, R. C. Lunardi, R. A. Michelin, A. M. Orozco, and A. F. Zorzo, "Lightweight IPS for port scan in OpenFlow SDN networks," in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2018, pp. 1–6.

[6] "Centro de estudos resposta e tratamento de Incidentes de Segurança no Brasil," *CERT.br - Estatísticas*.

[7] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[8] I. A. Valdovinos, J. A. Pérez-Díaz, K.-K. R. Choo, and J. F. Botero, "Emerging DDoS attack detection and mitigation strategies in software-defined networks: Taxonomy, challenges and future directions," *Journal of Network and Computer Applications*, vol. 187, p. 103093, 2021.

[9] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[10] F. Musumeci, A. C. Fidanci, F. Paolucci, F. Cugini, and M. Tornatore, "Machine-learning-enabled ddos attacks detection in p4 programmable networks," *Journal of Network and Systems Management*, vol. 30, pp. 1–27, 2022.

[11] F. Musumeci, V. Ionata, F. Paolucci, F. Cugini, and M. Tornatore, "Machine-learning-assisted DDoS attack detection with P4 language," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[12] M. Aamir, S. S. H. Rizvi, M. A. Hashmani, M. Zubair, and J. Ahmad, "Machine learning classification of port scanning and DDoS attacks: A comparative analysis," *Mehran University Research Journal Of Engineering & Technology*, vol. 40, no. 1, pp. 215–229, 2021.

[13] H. Taud and J. Mas, "Multilayer perceptron (MLP)," *Geomatic approaches for modeling land change scenarios*, pp. 451–455, 2018.

[14] Y. Zhai and X. Zheng, "Random forest based traffic classification method in SDN," in *2018 international conference on cloud computing, big data and blockchain (ICCBB)*. IEEE, 2018, pp. 1–5.

[15] R. Ritzkal, S. Suhadi, R. Amalia, Y. Afrianto, A. Triawan, S. Syafrial, and F. Fatimah, "K-nearest neighbor algorithm analysis for path determination in network simulation using software defined network," *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 4, pp. 2388–2400, 2023.

[16] S. Suthaharan and S. Suthaharan, "Support vector machine," *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*, pp. 207–235, 2016.

[17] G. Simsek, H. Bostan, A. K. Sarica, E. Sarikaya, A. Keles, P. Angin, H. Alemdar, and E. Onur, "Dropppp: a P4 approach to mitigating dos attacks in SDN," in *Information Security Applications: 20th International Conference, WISA 2019, Jeju Island, South Korea, August 21–24, 2019, Revised Selected Papers 20*. Springer, 2020, pp. 55–66.

[18] A. da Silveira Ilha, Â. C. Lapolli, J. A. Marques, and L. P. Gaspary, "Euclid: A fully in-network, P4-based approach for real-time DDoS attack detection and mitigation," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3121–3139, 2020.

[19] D. Ding, M. Savi, and D. Siracusa, "Tracking normalized network traffic entropy to detect DDoS attacks in P4," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 6, pp. 4019–4031, 2021.

[20] R. N. Carvalho, L. R. Costa, J. L. Bordim, and E. A. Alchieri, "Detecting ddos attacks on sdn data plane with machine learning," in *2021 Ninth International Symposium on Computing and Networking Workshops (CANDARW)*. IEEE, 2021, pp. 138–144.

[21] D. Barradas, N. Santos, L. Rodrigues, S. Signorello, F. M. Ramos, and A. Madeira, "FlowLens: Enabling Efficient Flow Classification for ML-based Network Security Applications." in *NDSS*, 2021.

[22] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM computer communication review*, vol. 38, no. 2, pp. 69–74, 2008.

[23] D. Ono, S. Izumi, T. Abe, and T. Suganuma, "A design of port scan detection method based on the characteristics of packet-in messages in openflow networks," in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 2020, pp. 120–125.

[24] "Open network operating system (ONOS) SDN controller for SDN/NFV solutions," *Open Networking Foundation (ONF)*.

[25] "P4Runtime Specification," *P4Runtime Specification," P4 - Language Consortium.*

[26] "CICFlowMeter," *Canadian institute for cybersecurity (CIC).*

[27] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018.

[28] "Stratum - Enabling the era of next-generation SDN," *Open Networking Foundation.*

[29] "SimpleSwitchGrpc - a version of SimpleSwitch with P4Runtime support." [Online]. Available: https://github.com/p4lang/behavioral-model/tree/main/targets/simple_switch_grpc

ANEXOS

Anexo A. Tablas con los resultados por ventana de tiempo

Métrica	**MLP**	**RF**	**KNN**	**SVM**
Precisión	0,996	0,997	0,996	0,996
Sensibilidad	0,953	0,903	0,950	0,948
Exactitud	0,955	0,913	0,954	0,952
F1-score	0,974	0,948	0,972	0,972

TABLA IX: Resultados ventana 5 segundos

Métrica	**MLP**	**RF**	**KNN**	**SVM**
Precisión	0,991	0,999	0,994	0,987
Sensibilidad	0,980	0,910	0,922	0,960
Exactitud	0,974	0,920	0,927	0,954
F1 - score	0,985	0,952	0,957	0,974

TABLA X: Resultados ventana 20 segundos

Métrica	**MLP**	**RF**	**KNN**	**SVM**
Precisión	0,997	1,000	0,998	0,998
Sensibilidad	0,955	0,974	0,876	0,949
Exactitud	0,958	0,976	0,887	0,951
F1 - score	0,976	0,986	0,933	0,973

TABLA XI: Resultados ventana 40 segundos

Métrica	**MLP**	**RF**	**KNN**	**SVM**
Precisión	0,998	0,999	0,999	0,998
Sensibilidad	0,936	0,858	0,860	0,940
Exactitud	0,940	0,867	0,871	0,943
F1 - score	0,966	0,923	0,924	0,968

TABLA XII: Resultados ventana 60 segundos