ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Sistema de Seguridad inteligente para monitoreo de ataques en una WSN para ambientes de agricultura de precisión.

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Telemática

Presentado por:

Kevin Gregorio Palacios Guzmán

Andrés Alejandro Cevallos Cevallos

GUAYAQUIL - ECUADOR

Año: 2024

DEDICATORIA

Kevin Gregorio Palacios Guzman

Deseo dedicar este proyecto a mis padres quienes me otorgaron la oportunidad de estudiar una carrera universitaria, a mi familia por todos los momentos que hemos pasado juntos y estar a mi lado en todo el camino recorrido hasta culminar mis estudios.

Andrés Alejandro Cevallos Cevallos

Dedico este proyecto a mi madre amada que siempre me apoyó y me guió en cada paso que di durante mi vida. Me apoyó constante e incondicionalmente y nunca dejó de creer en mi. Fue mi motor para seguir adelante incluso cuando creí que no podía. Sé que desde donde sea que se encuentre, ella estará orgullosa de saber que voy a graduarme.

AGRADECIMIENTOS

Andrés Alejandro Cevallos Cevallos

Quiero agradecer a mi familia, a Dios y a mis amigos por su apoyo incondicional. Ellos me brindaron la fuerza necesaria para poder culminar mi carrera y convertirme en un profesional. Agradezco a mi madre que estuvo junto a mi en cada paso que di y sé que desde el cielo ella me acompaña. A mi amiga Ginger que me ayudó a superar este desafío académico y juntos nos apoyábamos cuando mas lo necesitamos. Por último, quiero agradecer a mis profesores de carrera por su apoyo diario.

DECLARACIÓN EXPRESA

”Nosotros Kevin Gregorio Palacios Guzmán y Andrés Alejandro Cevallos Cevallos acordamos y reconocemos que: La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá al autor o autores, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor del autor o autores. La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada por nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de nuestra innovación, de ser el caso. En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique los autores que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.”

Guayaquil, viernes 14 de febrero del 2025.

Firmado electrónicamente por:
KEVIN GREGORIO
PALACIOS GUZMAN

Kevin Gregorio Palacios Guzmán

Firmado electrónicamente por:
ANDRES ALEJANDRO
CEVALLOS CEVALLOS

Andrés Alejandro Cevallos Cevallos

EVALUADORES

Firmado electrónicamente por:
MARIA ISABEL
MERA COLLANTES

PhD María Isabel Mera

Profesor de la Materia

Firmado electrónicamente por:
NESTOR XAVIER
ARREAGA ALVARADO

Mgtr Néstor Arreaga

Tutor del Proyecto

RESUMEN

En la agricultura de precisión, garantizar la seguridad en redes de sensores inalámbricos (WSN) es esencial para mantener la integridad de los datos del suelo y optimizar la productividad agrícola. Este proyecto desarrolla un sistema inteligente para detectar amenazas cibernéticas destinadas a este tipo de redes, como ataques DoS o denegación de servicio e inyección de datos, y mitigar su impacto.

Para ello, se analizó el protocolo ESP-NOW, utilizado por la WSN de nuestro cliente, y el tráfico entre dispositivos ESP8266 en la WSN con el fin de determinar variables para predecir ataques DoS. Se implementaron medidas de seguridad, como una lista de acceso basada en direcciones MAC autorizadas y la validación de mensajes mediante HMAC-SHA256. Un microcontrolador configurado como sniffer capturó y analizó el tráfico, permitiendo la creación de un modelo de detección que incluyó la recopilación y preprocesamiento de datos, selección de características, entrenamiento y validación en datasets recopilados y tráfico en vivo.

Los resultados muestran que el modelo clasifica correctamente el tráfico de red hacia el nodo receptor, diferenciando entre tráfico normal y ataques DoS con una precisión del 99.9% bajo condiciones específicas.

Este enfoque mejora la seguridad y confiabilidad de las WSN en entornos agrícolas, incentivando su adopción al garantizar la integridad de los datos y respaldar decisiones basadas en información precisa. **Palabras Clave:** WSN, Ataques de Red, Agricultura de precisión, Inteligencia Artificial, ESP-NOW

ABSTRACT

In precision agriculture, ensuring the security of wireless sensor networks (WSN) is essential to maintain soil data integrity and optimize agricultural productivity. This project develops an intelligent system to detect cyber attacks targeting such networks, including DoS or denial of service attacks and data injection, and mitigate their impact.

To achieve this, the ESP-NOW protocol, used by our client's WSN, and the traffic between ESP8266 devices were analyzed to determine features that can predict DoS attacks. Security measures were implemented, such as an access list based on authorized MAC addresses and message validation using HMAC-SHA256. A microcontroller configured as a sniffer captured and analyzed network traffic, enabling the development of a detection model. This process involved data collection, preprocessing, feature selection, training, and validation using both collected datasets and live traffic.

The results show that the model accurately classifies network traffic destined for the receiving node, distinguishing between normal traffic and DoS attacks with 99.9% precision under specific conditions.

This approach enhances the security and reliability of WSNs in agricultural environments, encouraging their adoption by ensuring data integrity and supporting decisions based on accurate information.

Keywords: *WSN, Network Attacks, Precision Agriculture, Artificial Intelligence, ESP-NOW*

ÍNDICE GENERAL

ABREVIATURAS

ESPOL	Escuela Superior Politécnica del Litoral
WSN	Wireless Sensor Networks
PMK	Primary Master Key
LMK	Local Master Key
LST	Laboratorio de Sistemas Telemáticos
MAC	Medium Access Control
HMAC	Hash Message Authentication Code
DoS	Denial of Service
IoT	Internet of Things
IA	Inteligencia Artificial
IF	Isolation Forest
RF	Random Forest

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

Hoy en día, la agricultura de precisión se ha transformado en un recurso clave para mejorar la eficiencia de la producción agrícola y se considera una garantía para la sostenibilidad de los recursos. Al integrarse con las tecnologías computacionales, la agricultura de precisión contribuyó al incremento su rentabilidad, así como a una mejora de los rendimientos, cantidad y calidad de los terrenos cultivados y los sembradíos (Cuesta and Solís, 2022).

La adopción de una Wireless Sensor Networks (WSN) para el monitoreo en agricultura permite a los dueños de plantaciones o parcelas obtener datos en tiempo real sobre las condiciones del suelo, clima y cultivos, lo que facilita la toma de decisiones fundamentadas. No obstante, el empleo de estas tecnologías también presenta desafíos importantes en términos de seguridad, dado que las WSN son susceptibles a diversos tipos de ataques cibernéticos que pueden afectar la integridad y disponibilidad de los datos (Romero Amondaray et al., 2020).

En un ambiente de agricultura una WSN está conformada por nodos sensores encargados de medir variables ambientales y del suelo para luego enviar los datos hacia un nodo receptor y finalmente a un Gateway o dispositivo de enlace a internet donde los datos recopilados serán manejados según su utilidad. Entre las características básicas de los nodos sensores están: tamaño reducido, bajo consumo de energía, capacidad de procesamiento limitada, memoria de almacenamiento limitada y la capacidad de comunicarse con otros dispositivos vía Wireless. Estas propiedades hacen que los nodos sensores tengan un tiempo de vida mayor con respecto a dispositivos tradicionales. No obstante, debido a estas características, no cuentan con mecanismos avanzados de seguridad ya que el dispositivo no cuenta con

los recursos para utilizarlos. (Romero Amondaray et al., 2020).

En este proyecto nos hemos propuesto desarrollar un sistema de seguridad inteligente capaz de monitorear y detectar, en cierta medida, ataques de red tipo DoS e inyección de datos en una WSN aplicada a la agricultura de precisión. A través del uso de algoritmos de inteligencia artificial se podrá identificar que tráfico de red es sospechoso o anómalo y, en función del contexto adecuado, generar una notificación y catalogarlo como un ataque. Este sistema de seguridad se integra al proyecto existente WSN Garden ubicado en el área verde del LST de la ESPOL, el cual se enfoca en la recolección de datos de humedad de suelo para determinar y/o predecir el momento adecuado para el riego. La activación del sistema de riego se basa en los valores detectados por los sensores, controlando una válvula que regula el flujo del agua.

1.1 Definición de la problemática

En la agricultura de precisión es imperativo tener conocimiento de las condiciones específicas del suelo, ya que se puede aprovechar una parcela para tener distintos tipos de cultivo sin afectar a los mismos (Castillo, 2023).

La falta de mecanismos de seguridad robustos en estos sistemas puede resultar en la pérdida de datos vitales, influir en la toma de decisiones de riego de forma negativa y, en última instancia, afectar la productividad y sostenibilidad de las operaciones agrícolas, por lo tanto es necesario tomar precauciones como lo es el desarrollar un sistema de seguridad inteligente que sea capaz de detectar este tipo de amenazas y actuar en base a ello.

Los nodos sensores dentro de una red de sensores inalámbricos presentan limitaciones físicas inherentes como:

- Eficiencia energética.- La vida útil de las baterias empleadas depende de la configuración establecida para los nodos. Existen soluciones para incrementar la duración de las baterias, librerías tipos ”sleep” las cuales configuran estados de activación para los microcontroladores, permitiendo así disminuir el consumo

energético en intervalos de tiempo donde el microcontrolador se encuentre inactivo.

- Tolerancia a fallos.- Es esencial desarrollar mecanismos que permitan detectar y notificar cuando un nodo haya agotado su energía, ya que en este estado no podrá recopilar ni transmitir información. También hay que saber diferenciar cuando un nodo esta funcionando de manera extraña al recibir información incoherente, sospechando así de una posible avería.
- Adversidades del entorno.- En climas muy fríos o muy calientes se tendrá que revisar las especificiaciones del fabricante para el microcontrolador utilizado con el fin de verificar las temperaturas de operabilidad permitidas, ya que puede llegar a sobrecalentarse el microcontrolador en temperatura altas y podría funcionar de forma no deseada entregando datos imprecisos.

Además de limitaciones físicas también existen vulnerabilidades y riesgos cibernéticos. Las WSN son susceptibles a varios tipos de ataques, aquí algunos de ellos:

1. Ataques activos:
    - Ataques DoS.- Denial of Service o denegación de servicio, es donde el atacante interfiere con el flujo de datos en la red ya sea incrementando el tráfico con muchos paquetes o enviando paquetes de gran tamaño para que los mensajes tarden en llegar a su destino o simplemente no lleguen (Sharma et al., 2022).
    - Ataques de interferencia.- El atacante interfiere con las señales inalámbricas para evitar el envío de paquetes (Lata et al., 2021).
2. Ataques pasivos:
    - Monitoreo y escucha.- El término monitoreo hace referencia a la observación activa de sistemas, redes o dispositivos para recopilar información sobre su estado, rendimiento o actividad. En general, el monitoreo es legítimo y se realiza para garantizar el funcionamiento adecuado de los sistemas, prevenir problemas, y detectar anomalías o ataques, sin embargo, también puede funcionar como una recopilación de información necesaria para realizar ataques activos. Escucha, se

refiere a la captura no autorizada de paquetes con el propósito de obtener información confidencial.

Según un estudio realizado por Puente Fernández (Puente Fernández, 2021), las redes de sensores inalámbricos (WSN) son vulnerables a ataques, por ejemplo: interceptación de datos, la manipulación de información y la denegación de servicio. Esto es debido a que estos sistemas trabajan con dispositivos de bajo consumo energético y capacidades de cómputo reducidas, lo cual dificulta la implementación de mecanismos de seguridad integrados. Como resultado, la integridad de la información puede verse comprometida, lo que afecta la precisión de las decisiones o acciones automatizadas en el sector agrícola. En el contexto de la agricultura de precisión, esto involucra una pérdida de control sobre el monitoreo del suelo, donde la obtención de datos exactos es fundamental (López Capó, 2024).

1.2 Justificación

Desde hace varios años en el sector agroindustrial ha emergido con fuerza la agricultura de precisión, un nuevo paradigma que aprovecha al máximo las condiciones del medio ambiente y de los cultivos, así como de los recursos materiales disponibles, para desarrollar este sector tan sensible (Tobar, 2022).

Las redes de sensores inalámbricos benefician a la agricultura de precisión de muchas maneras. Por ejemplo al proporcionar datos en tiempo real sobre variables clave como la humedad del suelo, la temperatura y la calidad del aire. Esto permite a los agricultores tomar decisiones informadas, automatizar, optimizar el riego y el uso de fertilizantes, para así mejorar la eficiencia en el manejo de los cultivos.

La implementación de un sistema de seguridad inteligente dentro de una WSN es crucial por varias razones. En primer lugar, la integridad de los datos recolectados es fundamental para la toma de decisiones. Un ataque que altere los datos puede resultar en malas decisiones agronómicas, ocasionando el rechazo de lotes debido a la perdida de control de calidad, esto incurre en perdidas económicas. En segundo

lugar, la seguridad de las WSN no solo protege a los agricultores, sino también a toda la cadena de cultivos, asegurando que los productos lleguen al consumidor final sin compromisos (López et al., 2022).

Por lo tanto, es necesario la implementación de un sistema de monitoreo de ataques que, además de detectar y notificar sobre los ataques en tiempo real, sea capaz de ajustarse a las particularidades del entorno agrícola determinado.

1.3 Objetivos

El objetivo general de este proyecto es: Monitorear los ataques que se pueden producir en una WSN para un ambiente de agricultura y alertar a los dueños de la información. Este objetivo general se cumplirá mediante los siguientes objetivos específicos:

1. Analizar la WSN existente junto al LST y completarla incrementando 2 nodos.
2. Diseñar y desarrollar un sistema de monitoreo inteligente usando inteligencia artificial para monitorear la anomalía de los datos.
3. Configurar un router inalámbrico y extensores para LST.

1.4 Alcances y Limitaciones

Para el alcance de nuestro proyecto nos centraremos en el desarrollo de un modelo de inteligencia artificial que sea capaz de detectar tráfico anómalo dentro de una red de sensores inalámbricos WSN que trabaja en un entorno de agricultura. El sistema estará diseñado específicamente para el monitoreo y análisis de datos de la humedad del suelo provenientes de los nodos sensores entregados por la actual WSN dentro del LST. Los datos anómalos están definidos como producto de ataques maliciosos de red, y nos enfocaremos en dos tipos:

1. **Denegación de servicio (DoS):** Este tipo de ataque lo definiremos como un ingreso irregular de paquetes, en este caso un incremento de estos. Entonces

lo definiremos como tráfico anómalo y será determinado por el modelo de inteligencia artificial. Adicional detectaremos el tamaño de los paquetes, ya que trabaja con dispositivos de bajo consumo y capacidad de procesamiento limitada.

2. **Valores incoherentes:** Estos son valores que se escapan de un rango lógico, al ser un sistema de nodos que trabajan todos los días y dependiendo de su configuración varias veces al día, se toma en cuenta también la frecuencia y duración del riego.

   Se han incluido mecanismos básicos de seguridad como una lista de MAC permitidas, mas conocido como AllowList, en esta lista se encontraran todas las direcciones MAC de cada nodo sensor pertenciente físicamente a la WSN. También se añadirá un mecanismo de identificacion de mensajes como un hash generado por el mensaje y una clave secreta.

Los recursos tecnológicos utilizados para este proyecto, así como la toma de datos y las pruebas, se realizarán en la red de sensores inalámbricos (WSN) instalada actualmente en el jardín alrededor del Laboratorio de Sistemas Telemáticos (LST). El estado en que se encuentre dicha red influirá directamente en la velocidad de las pruebas y, por ende, en el desempeño de nuestro sistema de monitoreo, ya que es fundamental que la WSN esté operativa para garantizar la recolección de datos. Si bien es posible trabajar con datos previamente recopilados, es necesario una recolección, ajuste y retroalimentación continua para crear un sistema automatico y funcional dentro la WSN.

1.5 Marco teórico

1.5.1 Redes de Sensores Inalámbricas

De acuerdo al estudio elaborado (López et al., 2022), las WSN son un conjunto de sensores que están difundidos en un terreno con el objetivo de monitorear condiciones físicas o ambientales y cuyas señales inalámbricas son recolectadas por un nodo a través de un dispositivo de borde.

El nodo sensor inalámbrico tradicional está conformada pr cuatro elementos: un módulo de sensores, un módulo de procesamiento con memoria, un módulo transceptor y una unidad de energía. Dentro de una red, dichos nodos tienen la capacidad de recolectar y procesar información, además de comunicarse entre sí. Las redes de sensores inalámbricos tienen como principales tienen como principal propósito garantizar la privacidad o confidencialidad, mantener la integridad de los datos, asegurar la autenticación y garantizar la disponibilidad del sistema (Keerthika and Shanmugapriya, 2021).

Las redes de sensores inalámbricos están ampliando su uso en diferentes sectores socioeconómicos, como la medicina, la ganadería, la agricultura, la industria, la manufactura y el transporte (Paavola and Leiviska, 2010), (Yu et al., 2013), (Peñín et al., 2017). No obstante, este incremento en su uso también resulta en desafíos significativos en términos de vulnerabilidades de seguridad, debido a la variedad de sus aplicaciones. En relación con las redes cableadas, las WSN presentan ventajas como la eliminación de los costos asociados al cableado extenso, la facilidad para reemplazar dispositivos averiados y la flexibilidad para modificarlos, en vista que no dependen de una infraestructura física compleja más allá del equipo necesario (Mendoza et al., 2020).

De acuerdo a los autores Kandris y demás (Kandris et al., 2020), se redacta que existen varias aplicaciones de las WSN que se utilizan ya de forma desarrollada o se encuentran aún en fases incipientes de desarrollo. Las aplicaciones de las WSN se clasifican según la naturaleza de su uso en seis categorías principales, que son las siguientes: militar, medioambiental, sanitaria, flora y fauna, industrial y urbana.

1.5.2 Internet de las Cosas (IoT)

El Internet de las cosas (IoT) es la tecnología que facilita la conexión de objetos físicos cotidianos a Internet. Esta tecnología abarca desde artículos domésticos comunes, como bombillas inteligentes, hasta herramientas de atención médica, dispositivos médicos, ropa, accesorios inteligentes, e incluso infraestructuras en ciudades inteligentes (University Carlemany, 2024).

Es una tecnología que enlaza prácticamente todo lo que hacemos, creando un ecosistema interconectado de dispositivos y máquinas. Esta conectividad permite a los usuarios monitorear el entorno e interactuar con equipos destinados como actuadores desde cualquier lugar con conexión a internet. Entre sus numerosas ventajas se encuentran el aumento de la productividad y la disminución del impacto ambiental. Los sensores IoT permiten un uso más eficiente de agua y fertilizantes, disminuyendo la contaminación por escorrentía. Se evita el uso innecesario de pesticidas, reduciendo su impacto en el suelo y los ecosistemas. (Mouha et al., 2021).

El IoT ha logrado desplazar a otras tecnologías adyacentes debido a su prometedor futuro y capacidad para facilitar el análisis y estudio de diversos elementos. Se espera que la agricultura se fortalezca enormemente con el avance de la tecnología, especialmente en el ámbito del IoT. La agricultura de precisión es el nuevo término agregado al campo de la agricultura, con todos los procedimientos implementados, gestionados y simulados de manera impulsada por la tecnología. La incorporación de internet ha comenzado a revolucionar este campo al conectar dispositivos entre sí, ahora identificado como el Internet de las Cosas (Kour and Arora, 2020).

1.5.3 Protocolo ESP-NOW

Uno de los protocolos que se utilizó en el modelo es ESP-NOW. Según los siguientes autores (Pasic et al., 2021), ESP-NOW es un protocolo de comunicación Wi-Fi sin conexión definido por Espressif. Los datos de la aplicación en ESP-NOW se encapsulan en un cuadro de acción específico y luego se transmiten de un dispositivo Wi-Fi a otro sin necesidad de conexión. A diferencia de los protocolos tradicionales como el estándar WiFi IEEE 802.11, ESP-NOW se centra en una comunicación directa y eficiente entre dispositivos. Esta característica lo hace una opción valiosa para aplicaciones en el ámbito del Internet de las Cosas (IoT) y en sistemas de sensores. ESP-NOW puede funcionar con Wi-Fi y Bluetooth LE, y es compatible con las series de SoC ESP8266, ESP32, ESP32-S y ESP32-C. Se utiliza ampliamente

en electrodomésticos inteligentes, control remoto, sensores, entre otros. Tiene un tamaño máximo de payload de 250 bytes.

El protocolo ESP-NOW está basado en la capa de enlace de datos y la física, que reduce las cinco capas del modelo OSI a solo dos. De esta manera, los datos no necesitan transmitirse a través del resto de capas. Además, no se requieren encabezados de paquetes ni desempaquetadores en cada capa, lo que permite una respuesta rápida y reduce el retraso causado por la pérdida de paquetes en redes congestionadas. (Expressif, n.d.).

Ya que ESP-NOW opera en la capa de enlace, se basa en direcciones MAC para establecer la comunicación, sin necesidad de una dirección IP, router o AP. Para establecer una conexión o emparejamiento, es necesario configurar los roles de cada nodo o microcontrolador involucrado, tiene que existir al menos 1 nodo con el rol de Receptor y 1 nodo con el rol de Emisor y conocer la dirección MAC del nodo receptor.

ESP-NOW es vulnerable a varios ataques, entre ellos:

1. **Ataque de Saturación de Tráfico:** Un atacante puede enviar grandes cantidades de tráfico con la misma dirección MAC de los nodos transmisores, sobrecargando al receptor y causando pérdida de paquetes y reducción en la tasa de transmisión.
2. **Suplantación de MAC:** Un atacante envía paquetes desde un dispositivo con la dirección MAC de un nodo transmisor, generando información falsa o congestionando la red.
3. **Ataque de Jamming:** Un dispositivo emite en la frecuencia de 2.4 GHz para interferir y bloquear la comunicación.

1.5.4 Agricultura de precisión

La agricultura de precisión es una estrategia de gestión que permite abordar las variabilidades geográficas y temporales en los campos agrícolas que implica el uso de datos y tecnología contemporánea. Debido al rápido crecimiento de población

mundial, la agricultura de precisión se está volviendo cada vez más importante en la investigación agrícola contemporánea. La agricultura de precisión implica una estrategia de gestión que utiliza un conjunto de técnicas avanzadas de información, comunicación y análisis de datos en el proceso de toma de decisiones (por ejemplo, aplicación de agua, fertilizantes, pesticidas, semillas, combustible, mano de obra, etc.), lo que ayuda a mejorar la producción de los cultivos y a reducir las pérdidas de agua y nutrientes y el impacto medioambiental negativo (Sishodia et al., 2020).

Para 2050, la cantidad de alimentos producidos a nivel mundial crecerá al menos un 70%. Esta es una tarea difícil porque ejerce una mayor presión sobre los recursos ya escasos y el medio ambiente. Por lo tanto, la agricultura de precisión es esencial para maximizar la producción utilizando menos insumos de todo tipo de manera más efectiva, reduciendo los impactos adversos en el medio ambiente y asegurando la sostenibilidad (Karunathilake et al., 2023).

En la agricultura de precisión las tecnologías utilizadas se encuentran en constante evolución debido a su amplio uso aparte del mencionado. El análisis de Big Data, la inteligencia artificial (IA), el Internet de las Cosas (IoT) y el aprendizaje automático podrían ser aplicados, optimizados y combinados para facilitar la toma de decisiones de gestión en base a información precisa (Adapt, 2023).

Una de las maneras en que la agricultura de precisión es utilizada en Ecuador es mediante el uso de sensores, dispositivos utilizados para medir temperatura, conductividad eléctrica o humedad del suelo. Los datos recopilados permiten determinar una cantidad óptima de riego necesaria para distintos tipos de plantaciones, determinar el estado del suelo o monitorear el uso adecuado de pesticidas empleados. Adicionalmente, la agricultura de precisión contribuye a mejorar la eficiencia de la fertilización, los sensores pueden medir la cantidad de nutrientes en el suelo y esta información obtenida nos permite conocer la cantidad optima de fertilizante a utilizar. (Palacios et al., 2024).

En el Ecuador, las redes de sensores inalámbricos empleados para mejorar

la agricultura de precisión, son usadas para monitorear la humedad relativa, la temperatura, la humedad del suelo, la luz y la lluvia. Además, posibilitan la reducción del gasto agua y de la utilización de pesticidas, tan perjudiciales para la salud humana, de los cultivos y de los suelos. Por último, en los cultivos ayudan a controlar la temperatura y la humedad de cada planta, para mantener el estado ideal de la misma y entregar un buen producto para el consumo humano (Palacios et al., 2024).

1.5.5 Aprendizaje por Refuerzo (Reinforcement Learning)

El aprendizaje por refuerzo (RL) es un enfoque de aprendizaje encargado de optimizar una política en función de un objetivo especifico a través de la interacción con un entorno. Entonces, en el proceso, un agente es encargado de observar el estado del entorno y de recibir una señal de recompensa en relación de sus acciones, las cuales también modifican dicho estado (Sutton y Barto, 2018). El RL se originó a partir de la convergencia de ideas resultantes de la inteligencia artificial, la ciencia cognitiva y la neurociencia. Su propósito es elegir acciones que permitan tomar decisiones con el fin de maximizar las recompensas futuras en la mayor medida posible (Matsuo et al., 2022).

A diferencia del aprendizaje supervisado, el aprendizaje por refuerzo no utiliza ejemplos etquetados que identifiquen comportamientos correctos o incorrectos. No obstante, también se distingue del aprendizaje no supervisado, ya que en vez de identificar patrones ocultos en los datos, el RL es basado en un proceso de prueba y error y en una función de recompensa para mejorar su desempeño.

El RL se basa en la interacción entre el agente, su entorno y un objetivo específico que alcanzar. Esta relación existente suele describirse en la literatura mediante el Proceso de Decisión de Markov (MDP). Un agente de RL realiza el aprendizaje a través de la interacción con el respectivo entorno, utilizando el estado actual para determinar sus acciones y poder recibir las recompensas o sanciones de acuerdo a sus decisiones. A través del tiempo, modifica su comportamiento para identificar acciones que le permitan alcanzar un objetivo específico en relación a las recompensas. En los procesos de decisión de Markov, el espacio de estado

representa toda la información disponible en base al entorno, mientras que el espacio de acción abarca todas las posibles acciones que el agente puede ejecutar en un determinado estado (IBM, 2024).

1.5.6 NodeMCU ESP-8266

Según los autores (TORRES VENTURA et al., n.d.), el nodo ESP8226 es un dispositivo IoT que recibe el comando etiquetado start/stop. Es un microcontrolador con un integrado esp8266 manufacturado por expressif basado en un CPU RISC de 32 bits, Xtensa LX106 con un reloj de 80 MHz, RAM de 64 KB, memoria flash QSPI 512 KB. La comunidad de desarrolladores se ha beneficiado ampliamente por el módulo ESP8266 debido a su precio accesible y a la robustez de su soporte de software. Ha sido un elemento clave en el desarrollo de proyectos IoT, automatización del hogar, control remoto y muchas otras aplicaciones.

1.5.7 Ataque de Man in the Middle (MITM)

Es un tipo de ciberataque en el que un actor malicioso intercepta, modifica o se inserta en la comunicación entre dos partes, sin que ninguna de ellas lo note. El objetivo principal suele ser espiar, robar información confidencial o manipular los datos transmitidos. El atacante se posiciona entre dos partes (por ejemplo, entre un usuario y un servidor) y actúa como un intermediario. A menudo, las víctimas creen que están comunicándose directamente entre sí, pero en realidad, el atacante está controlando o monitoreando la comunicación. Entre los métodos comunes de un ataque están la suplantacion de ARP, en la cual el atacante engaña a los dispositivos en una red local para asociar su dirección MAC con la dirección IP de otra máquina, como el router; intercepción de wifi, donde se crean redes wifi falsas para que los usuarios se conecten y poder interceptar el tráfico; DNS Spoofing, la cual redirige a los usuarios a sitios web falsos modificando las respuestas del servidor DNS (Gonzaga and Sampaio, 2020).

1.5.8 Ataque Backdoor

Un ataque de backdoor es un tipo de ciberataque que permite acceder a un sistema informático, red o aplicación, evitando los mecanismos normales de autenticación o seguridad. Este tipo de acceso suele ser creado por atacantes para mantener el control del sistema a largo plazo y realizar actividades maliciosas sin ser detectados. Para ello, un atacante instala un software, modifica configuraciones o introduce código que crea una ”puerta trasera” en el sistema. Esto le permite volver a acceder al sistema sin necesidad de las credenciales legítimas o sin pasar por los controles de seguridad implementados (Gao et al., 2020).

1.5.9 Ataque de Denegación de Servicio (DoS)

Un ataque de denegación de servicio (DoS) es un tipo de ciberataque en el que un atacante intenta hacer que una computadora u otro dispositivo se vuelva inaccesible para sus usuarios legítimos, alterando su funcionamiento habitual. Estos ataques funcionan normalmente al sobrecargar el dispositivo objetivo con una gran cantidad de solicitudes, hasta el punto en que no puede procesar el tráfico regular, lo que impide el acceso a otros usuarios. Una característica de los ataques DoS es que se llevan a cabo utilizando una sola computadora (NCSC, 2016).

Los ataques DoS suelen ser de dos categorías:

1. **Ataques de desbordamiento de búfer:** Un ataque de desbordamiento de búfer en la memoria puede llevar a que una máquina agote el espacio del disco duro, la memoria o el tiempo de CPU. Esto suele resultar en un rendimiento lento, fallos del sistema u otros efectos negativos en el servidor, causando finalmente una denegación de servicio (Cloudfare, n.d.).

2. **Ataques de inundación:** Al enviar una cantidad excesiva de paquetes a un servidor objetivo, un atacante puede superar su capacidad, causando una denegación de servicio. Para que los ataques de inundación DoS sean efectivos, el atacante suele necesitar más ancho de banda disponible que el servidor objetivo.

.

1.5.10 HMAC-SHA-256

HMAC-SHA-256 es un tipo de algoritmo hash con clave que se construye a partir de la función hash SHA-256 y se usa como código de autenticación de mensajes basado en hash (HMAC). El proceso HMAC mezcla una clave secreta con los datos del mensaje, aplica un algoritmo hash al resultado con la función hash, mezcla ese valor hash con la clave secreta de nuevo y, a continuación, aplica la función hash una segunda vez. El hash de salida tiene una longitud de 256 bits (Microsoft, n.d.).

Este mecanismo es útil en sistemas donde la seguridad y la verificación de la autenticidad de los datos son cruciales, como en las redes de sensores inalámbricos (WSN), en donde se usan los HMACs para asegurar la comunicación entre nodos y gateways.

1.6 Estado del arte

De acuerdo a (Prodanović et al., 2020), hoy en día, los sensores se utilizan para la recopilación rápida de datos en tiempo real en la agricultura mediante redes de sensores inalámbricos (WSN). Los datos recopilados de esta manera muestran un cambio inmediato que permite al sistema monitorear los parámetros agrícolas en tiempo real. El acceso a una gran cantidad de datos generados de diferentes fuentes en poco tiempo genera dificultades para tomar la mejor decisión. El documento busca desarrollar un modelo de redes de sensores de seguridad que funcione de manera independiente de la infraestructura de comunicaciones, ofreciendo seguridad punto a punto y aplicándose en actividades humanas que necesitan proteger los datos recopilados para la gestión. Sin embargo, se concluyó que los competidores maliciosos pueden agregar o modificar datos medidos e interrumpir la aplicación y el proceso de producción o prueba.

En (Islam et al., 2021), los autores establecen que debido a las configuraciones de construcción simples y capacidad computacional limitada de los nodos sensores en

una WSN, estos son vulnerables a ataques de seguridad. Los ataques de denegación de servicio (DoS) son uno de los más frecuentes entre ellos. Los ataques DoS se utilizan para interrumpir el acceso a los recursos de una red. Generalmente, esto se logra inundando la máquina o red objetivo con tráfico innecesario, de modo que no pueda responder a las solicitudes legítimas de los usuarios autorizados. Los autores se enfocan en los ataques DoS a WSN y sus técnicas de prevención. Ellos llegan a la conclusión de que la mayoría de los mecanismos de defensa contra ataques DoS se centran en métodos como reconocimientos, cifrado, autenticación, etc. Estos enfoques son buenas soluciones para mitigar los ataques DoS en WSN; sin embargo, consumen recursos.

Los autores (Lata et al., 2021) mencionan que las redes de sensores inalámbricos (WSN) son una de las principales tecnologías necesarias para la implementación de la arquitectura de Internet de las cosas (IoT). Las WSN son las redes utilizadas para la comunicación entre sensores y transceptores de radio. En su estudio los autores cubren todos los ataques a WSN, detección y medidas preventivas para la integración de WSN e IoT. Además, proporcionan una guía para defender la IoT contra dichos ataques. Entre los ataques proporcionados están los ataques activos, ataques pasivos, ataques a la capa física, ataques a la capa de enlace, ataques a la capa de red, ataques a la capa de transporte y ataques a la capa de aplicación.

De acuerdo a (Triantafyllou et al., 2019), los autores presentan una descripción general de una arquitectura de sistema de monitoreo de agricultura inteligente. Ellos presentan un sistema de monitoreo de agricultura de precisión que se basa en detectar parámetros agrícolas, ubicar sensores, recopilar y enviar datos del campo a la estación de control para la toma de decisiones, actuar según la información detectada y mostrar los resultados al agricultor mediante una aplicación. El modelo sigue el estándar ISO/IEC 7498-1 (Modelo OSI), con un sistema de comunicación estructurado en siete capas de abstracción.

En (Abunadi et al., 2022) se presenta un marco para integrar la computación

en la niebla con la agricultura inteligente, optimizando el control del tráfico de red. El sistema propuesto monitorea de forma eficiente la información redundante, evitando el uso excesivo del ancho de banda, y limita las retransmisiones frente a acciones maliciosas, maximizando el uso de los recursos de la red. Además, establece una conexión confiable entre los sensores agrícolas mediante nodos de niebla para mejorar la seguridad y prevenir comunicaciones maliciosas. Los experimentos basados en simulaciones mostraron que este enfoque supera a otros trabajos en eficiencia energética, seguridad y conectividad de la red (Abunadi et al., 2022).

En (Bhasker and Murali, 2020), se plantea que las WSN ofrecen soluciones eficaces para diversos tipos de aplicaciones de gran tamaño en tiempo real. En la actualidad, las personas están mirando hacia entornos en la nube o más conocido como cloud para el procesamiento y almacenamiento de información de las WSN de acuerdo a sus necesidades. Los datos relacionados al riego son almacenados y procesados mediante tecnología en la nube, posibilitando que profesionales del sector agrícola accedan a la información en tiempo real. La principal ventaja de este enfoque es que proporciona un riego adecuado de los cultivos, por esto, se señala que el sistema de gestión de riego agrícola entrega una solución que permite optimizar el rendimiento de los cultivos y a su vez utiliza de manera eficiente las reservas de agua dentro de un campo sin fácil acceso al recurso hídrico.

Un trabajo investigativo desarrollado por estudiantes de la ESPOL (Gutiérrez Sánchez, Sánchez Sánchez, et al., 2022) en Ecuador, se centra en la problemática del ”Recinto el Rosal”, en esta localidad habitan comuneros que se encargan de supervisar la siembra y cosecha de cacao debido a la violación de la normativa dictaminada por la autoridad competente de Ecuador el MAGAP (Ministerio de Agricultura, Ganadería, Acuicultura y Pesca) respecto a la separación vegetal entre una planta y otra, relieve irregular que poseen las zonas de sembrados, amontonamiento de agua en ciertas zonas hídricas, gestión ineficiente o derroche de recursos. Por lo tanto, se ha planteado e implementado el diseño de una red de comunicación IoT que posee dispositivos finales y periféricos destinados a la adquisición y transmisión de datos de variables ambientales como humedad y

temperatura. Posteriormente, se implementó un prototipo a pequeña escala de un sistema de riego automatizado para contar con un sistema completo de actuadores y sensores.

Los autores mencionados (Palacios et al., 2024) plantean en el artículo científico que en el Ecuador, las redes de sensores inalámbricos empleados para mejorar la agricultura de precisión, permiten monitorear la humedad relativa, la temperatura, la humedad del suelo, la luz y la lluvia. Adicionalmente, estas tecnologías contribuyen a la reducción del gasto agua y de la utilización de pesticidas, cuyo impacto son tan perjudiciales para la salud humana, de los cultivos y de los suelos. Por último, en los cultivos ayudan a controlar la temperatura y la humedad de cada planta, para mantener el estado ideal de la misma y entregar un buen producto para el consumo humano.

El artículo (Haseeb et al., 2020) propone un marco de red de sensores inalámbricos (WSN) basado en IoT para la agricultura inteligente. El sistema emplea sensores agrícolas que capturan datos y seleccionan líderes de clústeres mediante una función de decisión multicriterio, optimizando la transmisión de datos con la relación señal a ruido (SNR). Además, incorpora seguridad en la transmisión de datos mediante un generador congruencial lineal. Las simulaciones demostraron mejoras en el rendimiento de la comunicación, incluyendo aumentos en el rendimiento de la red y reducciones en la pérdida de paquetes, latencia, consumo de energía y sobrecarga de enrutamiento.

El presente artículo desarrollado por los autores (Hossain et al., 2021) plantea que el aumento del flujo de información IoT en la agricultura de precisión (PA) podría comprometer la seguridad si no se implementa un mecanismo adecuado. La protección de la Propiedad Intelectual (PI) es fundamental para evitar la pérdida de ideas claves, como patrones de cultivo y datos generados por IoT. Se propone un sistema regulador de puerta de enlace con IoT y SDN que permite controlar dispositivos externos sin acceder a información sensible de la granja. Este sistema, gestionado desde un centro de control en la nube, busca optimizar la seguridad y

gestión de insumos agrícolas, incrementando la rentabilidad y reduciendo el riesgo de robo de PI.

A continuación, el artículo elaborado por los autores (Chaganti et al., 2022) ha desarrollado un marco de monitoreo de seguridad para la agricultura inteligente basado en la nube, diseñado para supervisar el estado de los dispositivos y detectar anomalías en sensores, mitigando ataques de seguridad mediante patrones de comportamiento. Además, se implementa un contrato inteligente en blockchain para almacenar y compartir información de anomalías de manera segura, ayudando a proteger otras granjas. El prototipo utiliza Arduino, ESP32, AWS y la red de pruebas Ethereum Rinkeby, y su evaluación mostró que detecta anomalías en tiempo real y notifica a otros nodos de la granja sobre posibles amenazas.

CAPÍTULO 2

2. METODOLOGÍA

Antes de presentar el hardware y software utilizado, mostraremos la WSN en la cual trabajamos tanto para pruebas como para resultados.

Figure 2.1: WSN - LST

En la figura 2.1 podemos ver la WSN actual, más adelante se detalla el estado de la WSN, por ahora trabajaremos con los nodos 1, 2 y 5. Esta WSN trabaja no solo en la recolección de valores de humedad de suelo sino también en la activación de una valvula que da paso al riego, la activacion de dicha valvula se realiza mediante una predicción por parte de un modelo Random Forest que determina de manera óptima tiempos de activación en base a la humedad recolectada y datos históricos.

En esta sección se detallan los componentes de hardware y software

empleados en la implementación de la red de sensores inalámbricos y el sistema de monitoreo y detección de ataques en el LST de la ESPOL.

2.0.1 Componentes de hardware

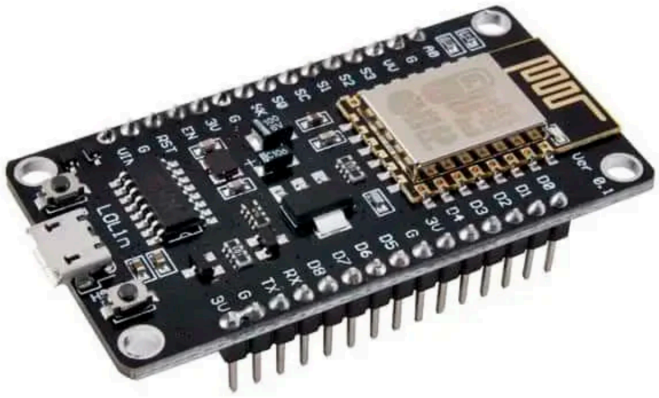


Figure 2.2: Microcontrolador NodeMCU ESP8266

Microcontrolador NodeMCU ESP8266: El NodeMCU es una plataforma de desarrollo de hardware de código abierto diseñada para proyectos de IoT (Internet de las Cosas). Está basada en el microcontrolador ESP8266, un chip con capacidad de conexión Wi-Fi desarrollado por Espressif Systems y en el módulo transceptor Wi-Fi ESP8266 y el chip convertidor USB CH340. Esta compacta placa de desarrollo y prototipos de código abierto es ideal para aplicaciones de IoT. El módulo Wi-Fi es compatible con el estándar 802.11 b/g/n a 2.4 GHz, tiene una pila TCP/IP integrada, una potencia de salida de 19.5 dBm, interfaz de datos (UART / HSPI / I2C / I2S / Control remoto Ir / GPIO / PWM) y antena PCB. También cuenta con un conector micro USB y un botón de reinicio. Es programable con Arduino IDE, incluye intérpretes para procesar comandos en lenguajes como LUA.

Figure 2.3: Sensor de Humedad HD-38

Sensor de Humedad hd-38: Es un sensor que mide humedad de suelo utilizado en la WSN, funciona en un rango operativo que va desde -25 a 85 grados Celsius. El sensor cuenta con una extension de hasta 1 metro para incrustar en el suelo a ser medido. Como uno de nuestros objetivos específicos es incrementar la cantidad de nodos existentes en la WSN, es necesario mantener el mismo modelo de equipos utilizados para evitar problemas de fabricantes o problemas no previstos debido al uso de distintos dispositivos de sensado.

Figure 2.4: Raspberry Pi V4

Raspberry Pi V4: El gateway utilizado en la WSN es una Raspberry Pi 4. Opera con un valor de 5 Voltios y una corriente de 2.5 Amperios para su funcionamiento básico. La comunicacion entre el gateway y el nodo receptor se logra mediante un

cable usb y el protocolo de comunicacion USB-serial. Es el equipo encargado del procesamiento de los datos de humedad recibidos. Su reducido tamaño y moderada capacidad de procesamiento lo vuelven una opción valiosa para actuar como Gateway. Entre sus especificaciones técnicas están que, a pesar de su tamaño compacto, ofrece un rendimiento notable, con un procesador de cuatro núcleos, hasta 8 GB de memoria RAM, conectividad Wi-Fi y Bluetooth, puertos USB y soporte para dos pantallas 4K.

2.0.2 Protocolos y estrategias

Herramientas digitales utilizadas por la WSN en funcionamiento, para las pruebas, librerías o guías para entrenar el modelo machine learning.

MAC Header	Category Code	Organization Identifier	Random Values	Vendor Specific Content	FCS
24 bytes	1 byte	3 bytes	4 bytes	7-257 [bytes]	4 bytes
Vendor Specific Content					
Element ID	Length	Organization Identifier	Type	Version	Body
1 byte	1 byte	3 bytes	1 byte	1 byte	0-250 [bytes]

Figure 2.5: Trama conocida paquetes ESP NOW

ESP-NOW: Es el protocolo establecido para la comunicación en la WSN existente de nuestro cliente debido a sus beneficios respecto a eficiencia energética. En la documentación publicada por Espressif, como podemos ver en la figura 2.5, encontramos los distintos segmentos que contienen los paquetes transmitidos dentro de la red, estos segmentos son utilizados para filtrar los paquetes pertinentes del resto de tráfico captado por el dispositivo sniffer.

Arduino: Entorno de desarrollo destinado a la programación de dispositivos arduino, entre otros, compatibles como la familia ESP. Su lenguaje está basado en C/C++, su interfaz permite añadir o buscar microcontroladores compatibles, así como también el manejo de librerías. Cuenta con mensajes de compilación y depuración, una gran cantidad de foros y ayuda en internet. Adicional, cuenta con una interfaz

serial interactiva y mensajes de depuración en tiempo de compilación y carga de código.

Lista de acceso: Este método consiste en crear una lista que contiene las direcciones MAC de únicamente los dispositivos involucrados o conocidos dentro de la red. Esta lista se usa como recurso en el código del nodo receptor para verificar la identidad del nodo emisor al enviar mensajes, esta funcionalidad nos permite ignorar los mensajes provenientes de posibles atacantes, sin embargo, aún existe la posibilidad de que algún atacante suplante la identidad de un nodo verificado, al enviar mensajes con una MAC autorizada y vulnere esta medida.

HMAC-SHA-256: Es el mecanismo de seguridad empleado para proteger la información de cada paquete transmitido en la WSN. Este mecanismo funciona mediante el uso de una clave secreta, conocida por el nodo receptor y todos los nodos emisores, junto con los valores del nombre del nodo y humedad; esto se lo convierte en un hash mediante la función SHA-256. El hash lo usaremos para confirmar la identidad del nodo emisor, ya que si cambia algún valor ya sea de humedad o la clave secreta, los HMAC no serán los mismos y se procederá a descartar esos valores. El resultado es una cadena de 32 bytes.

Un HMAC se puede usar para determinar si se ha alterado un mensaje enviado a través de un canal no seguro, siempre y cuando el remitente y receptor compartan la clave secreta el HMAC deberá ser el mismo. Si esto no ocurre, entonces se puede sospechar que hay un intruso en la red. El nodo receptor recalcula el HMAC con los valores recibidos y la clave secreta para luego comparar el HMAC que obtuvo en el mensaje del nodo emisor.

Sniffer de red: Un sniffer es una herramienta utilizada para interceptar y analizar el tráfico sin intervenir en la comunicacion de una red. Esta captura de paquetes permite observar en detalle el contenido de las transmisiones, como direcciones IP, puertos, protocolos y, en algunos casos, el contenido de los mensajes (Unir, 2023). Al utilizar el sniffing, el atacante puede capturar paquetes como tráfico Syslog, tráfico DNS, tráfico web, correo electrónico y otros tipos de tráfico de datos. Al capturar estos paquetes, un atacante puede revelar información como datos, nombre de usuario y contraseñas de protocolos como HTTP, POP, IMAP, SMTP,

FTP y Telnet. En el proceso de sniffing, un atacante se conecta a la red objetivo para rastrear los paquetes. Usando sniffers, que convierte la tarjeta de interfaz de red (NIC) del sistema del atacante en modo promiscuo, el atacante captura el paquete. Una vez que el atacante captura el paquete, puede descifrarlo para extraer la información. Los rastreadores pueden usarse para piratear un sistema o una red (Tuli, 2020). La Raspberry Pi será el dispositivo encargado de monitorear el trafico de red.

2.0.3 Métodos

Pruebas de mecanismo de seguridad: Primero, se implementó una lista de acceso con direcciones MAC autorizadas para evitar el reenvío de datos provenientes de dispositivos no reconocidos. Aunque es relativamente sencillo intercambiar datos entre dispositivos emisor y receptor, basta con contar con el equipo adecuado y acceder al formato de tramas oficial proporcionado por ESPRESSIF, además de conocer la MAC del receptor, para suplantar a un emisor y transmitir información.

Para reforzar la seguridad, utilizamos el algoritmo HMAC-SHA256, enviando un hash como identificador único de los mensajes. Esto implica que, incluso si un atacante obtiene la MAC del receptor y de un emisor autorizado, aún deberá generar y enviar un identificador válido para que los datos sean procesados. Sin embargo, si la verificación del hash falla, el paquete simplemente no será reenviado ni considerado, aunque la recepción inicial del paquete se haya completado.

El nodo receptor cumple su rol independientemente de los mecanismos empleados, aún recibe paquetes provenientes tanto de nuestros dispositivos autorizados como del resto de dispositivos que lleguen a ser emparejados. Esto expone a la red inalámbrica de sensores (WSN) a posibles ataques de denegación de servicio (DoS).

ESP8266 como sniffer de red: Inicialmente, consideramos utilizar una Raspberry Pi 4, empleando su tarjeta de red incorporada o un adaptador Wi-Fi, para monitorear el tráfico de la red WSN basada en ESP8266. Sin embargo, este enfoque resultó complicado debido a que ESP-NOW opera de manera directa en la capa 2, sin involucrar capas superiores, y tampoco sigue los estándares tradicionales de Wi-Fi.

Por esta razón, se optó por emplear uno de los ESP8266 como sniffer.

La librería ESP8266WiFi.h cuenta con un modo ”promiscuo” el cual permite al dispositivo escuchar las redes cercanas. Durante las pruebas logramos captar varios tipos de paquetes como los son beacon de los access point de alrededor y algunos transmitiendo data por otros dispositivos ESP de alrededor. Esto se determinó gracias al filtro 0x18FE34 el cual es característico de los dispositivos Espressif y la interpretacion de un campo llamado FrameControl, el cuál indica el tipo de paquete, tales como tipo Data, Management o Control. Entonces al captar paquetes del tipo Data pero de distinto contenido al esperado en nuestra red, pudimos llegar a la conclusión de que se trata de comunicación con otras redes ESP de alrededor del LST.

Código ESP8266: El código original de las ESP8266 fue modificado para permitir un mayor control sobre los paquetes enviados entre el emisor y el receptor. La estructura inicial de datos definida como:

```
struct Data
float humedad;
char Name[20];
char hmac[65];
Data;
```

se diseñó con este formato para prevenir desbordamientos y evitar pérdida de información. Sin embargo, al analizar los paquetes con nuestro sniffer, descubrimos que al asignar un tamaño fijo en memoria (20 o 65 bytes), se enviaba la totalidad del espacio asignado, rellenado con valores 0x00.

Esto permitió optimizar la estructura, reduciendo Name a 6 bytes. El formato para los nombres de los nodos maestros ahora es ”HUMxx”, donde ”xx” representa valores del 01 al 99. Aunque este formato solo necesita 5 bytes, se encontró que algunos datos llegaban con errores como ”HUM05 6yo _”. Para resolverlo, se añadió un byte extra como separador (rellenado con 0x00), asegurando que los datos enviados se interpreten correctamente.

La estructura final quedó definida así:

```
struct Data
float humedad;
```

char Name[6];
char hmac[32];

En el caso del HMAC-SHA256, se confirmó que el hash generado siempre ocupa exactamente 32 bytes, lo cual vuelve innecesario el espacio previamente reservado de 65 bytes.

Código nodo emisor: Primero, importamos las librerías que se muestran en la figura 2.6, librerías necesarias para la creacion del hash, iniciar la comunicación esp-now y manejar los modos WiFi.

Figure 2.6: Librerías utilizadas en el nodo emisor.

Entre las variables principales se encuentran las que se muestran en la figura 2.7 que son: SlaveAdd que hace referencia a la dirección MAC del dispositivo receptor, esta dirección debe ser conocida por todos los nodos emisores que deseen enviar datos dentro de la red.

```
// REPLACE WITH YOUR RECEIVER MAC Address
uint8_t SlaveAdd[] = { 0x48, 0x55, 0x19, 0x0B, 0xA2, 0xE6 };
const char *SECRET_KEY = "n3IoTiPensoL9";

typedef struct Data {
  float humedad;
  char Name[6];
  char hmac[32];
} Data;

Data data;
```

Figure 2.7: Estructura Data enviada.

SECRET_KEY, la cual es una clave secreta utilizada para generar el HMAC. Se define Data, que es una estructura que contiene:

1. **Humedad:** Valor obtenido por el procesamiento de la lectura análoga realizada por los nodos emisores gracias a los sensores hd-38.
2. **Name:** Una etiqueta para identificar cada nodo sensor.

3. **hmac:** Hash para garantizar la integridad y autenticidad de los datos.

Entre las funciones utilizadas para configurar los mecanismos básicos previamente definidos estan:

1. **generateHMAC:** Genera un HMAC utilizando SHA256 basado en: La clave secreta SECRET_KEY. El mensaje que incluye el nombre del sensor (Name) y el valor de la humedad. El resultado se almacena en el campo hmac de la estructura data.

2. **generarHumedad:** Se encarga de crear el mensaje a enviar, el cual contiene la humedad del suelo y el nombre del nodo emisor. Para pruebas y depuración se usaron salidas seriales de la información transmitida.

emisor §

```
22 Data data;
23
24 void generateHMAC(const char *message, char *outputHMAC) {
25   SHA256 sha256;
26   sha256.reset();
27   sha256.update((const uint8_t*)SECRET_KEY, strlen(SECRET_KEY));
28   sha256.update((const uint8_t*)message, 10);
29
30   //uint8_t hash[32];
31   sha256.finalize(outputHMAC, 32);
32 }
33 //(random(500, 81500) / 100.0)
34 void generarHumedad(){
35   //memset(payload, 0xBB, sizeof(payload));
36   strcpy(data.Name, "HUM01");
37   data.humedad = analogRead(sensorPin);
38   char message[10];
39   memcpy(message, data.Name, 6);
40   memcpy(message +6, &data.humedad, 4);
41
42   generateHMAC(message, data.hmac);
43
44 }
45 // Callback function called when data is sent
46 void OnDataSent(uint8_t *mac_addr, uint8_t status) {
```

Figure 2.8: Función generateHMAC y generateHumedad.

Para la configuracion o setup() como podemos ver en la figura 2.8, es recomendable agregar un delay antes de iniciar para evitar errores. Se procede a inicializar un serial para poder imprimir los mensajes que queremos enviar al gateway. Como la comunicacion será mediante esp-now, no es necesario anunciar el dispositivo como un punto de acceso y por lo tanto se configura el modulo WiFi en el modo ”Station” o ”WIFI_STA”. El resto de funciones propias de esp-now() son necesarias para iniciar el emparejamiento. esp_now_set_role(): Define si es un receptor o un emisor. esp_now_register_send_cb(): Es una función tipo callback par el rol de

emisor, es decir, se realiza la funcion especificada luego de realizar el envío.

emisor §

```
49
50 void setup() {
51   // Init Serial Monitor
52   delay(1000);
53   Serial.begin(9600);
54
55   // Set device as a Wi-Fi Station
56   WiFi.mode(WIFI_STA);
57
58   // Initilize ESP-NOW
59   if (esp_now_init() != 0) {
60     Serial.println("Error initializing ESP-NOW");
61     return;
62   } else {
63     Serial.println("Initialized ESP-NOW");
64   }
65
66   esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);
67   esp_now_register_send_cb(OnDataSent);
68
69   // Add peer
70   if (esp_now_add_peer(SlaveAdd, ESP_NOW_ROLE_SLAVE, 0, NULL, 0) != 0) {
71     Serial.println("Failed to add peer");
72     return;
73   } else {
```

Figure 2.9: Configuracion Emmisor

El resto de codigo es para fines prácticos y entorno de prueba, verificar que el emparejamiento se haya completado, definir el canal de comunicación.

Código nodo receptor: La configuración es similar con algunas diferencias como el rol establecido, en este caso ”Slave” o receptor. También cambia la función callback, se activa luego de haber recibido un mensaje. Esta es la funcion que usamos para procesar los mensajes recibidos y decidir si son validos o no.

```
void setup() {
  // Init Serial Monitor
  Serial.begin(9600);

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
  esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);
  esp_now_register_recv_cb(OnDataRecv);

}
```

Figure 2.10: Configuración inicial de nodo receptor.

Entre las funciones desarrolladas, se encuentran:

1. **verifyHMAC:** Es la función encargada de verificar el identificador de mensajes recibidos, recibe el mensaje que contiene el name y el valor de humedad y el HMAC recibido para luego volver a generar el HMAC haciendo uso de la clave secreta y comparar si coinciden retornando un valor booleano.

2. **isMACAuthorized:** Compara byte por byte la dirección MAC del dispositivo emisor con las encontradas en la lista de acceso y retorna un valor booleano.

3. **OnDataRecv:** Se ejecuta automáticamente cuando el ESP recibe datos. Realiza los siguientes pasos: Primero, verifica si la MAC del emisor está autorizada mediante isMACAuthorized; copia los datos recibidos en Data que definimos; reconstruye el mensaje original (nombre del sensor y humedad); verifica el HMAC recibido usando verifyHMAC. Si los datos son válidos, los imprime en el monitor serial. Si los datos son inválidos, los ignora.

4. **Callback OnDataSent:** Indica si el envío fue exitoso o fallido.

```
//verificar hmac
bool verifyHMAC(const char *message, const char *hmacRecibido) {
  SHA256 sha256;
  sha256.reset();
  sha256.update((const uint8_t*)SECRET_KEY, strlen(SECRET_KEY));
  sha256.update((const uint8_t*)message,10);

  char hash[32];
  sha256.finalize(hash, 32);

  Serial.println();
  Serial.print("HMAC recibido: ");
  for (int i = 0; i < 32; i++) {
    Serial.print(hmacRecibido[i], HEX);
  }
  Serial.println();

  Serial.print("HMAC generado: ");
  for (int i = 0; i < 32; i++) {
    Serial.print(hash[i], HEX);
  }
  Serial.println();

  Serial.print("Message generado (HEX): ");
  for (int i = 0; i < 10; i++) {
    Serial.print(message[i], HEX);
    Serial.print(" ");
  }
  Serial.println();
  return (memcmp(hash, hmacRecibido, 32) == 0);

}
```

Figure 2.11: Función verifyHMAC en nodo receptor.

```
bool isMACAuthorized(uint8_t *mac) {
  for (int i = 0; i < sizeof(WhiteList) / sizeof(WhiteList[0]); i++) {
    if (memcmp(mac, WhiteList[i], 6) == 0) {
      return true;  // MAC autorizada
    }
  }
  return false;  // MAC no autorizada
}

// callback function that will be executed when data is received
void OnDataRecv(uint8_t *mac, uint8_t *incomingData, uint8_t len) {
  if(!isMACAuthorized(mac)){
    Serial.println("MAC no autorizada.");
    return;
  }
  Data data;
  memcpy(&data, incomingData, sizeof(data));
  char message[10];
  memcpy(message, data.Name, 6);
  memcpy(message +6, &data.humedad, 4);

  if(verifyHMAC(message, data.hmac)){
    Serial.print("Data en formato HEX: ");
    for (int i = 0; i < len; i++) {
      Serial.print(incomingData[i], HEX);
      Serial.print(" ");
    }
    Serial.println();
    Serial.printf("Datos recibidos de: %s\n", data.Name);
    Serial.printf("Humedad: %.2f %%\n", data.humedad);
  } else{
    Serial.printf("Clave incorrecta, datos rechazados de: %s\n",data.Name);
    Serial.println();
```

Figure 2.12: Funciones isMACAuthorized y OnDataRecv del nodo receptor.

Código del sniffer: El ESP8266 tiene la capacidad de operar en modo promiscuo, lo que significa que puede escuchar algunos paquetes de datos transmitidos en un canal Wi-Fi, sin importar si están dirigidos a él.

Primero se configura el ESP en modo estación Wi-Fi (WIFI_STA) para que no intente conectarse a ninguna red; desactiva cualquier conexión activa con WiFi.disconnect() para asegurarse de que está libre para capturar tráfico; habilita el modo promiscuo con la función wifi_promiscuous_enable(true) y registra la función promiscuous_callback como el manejador para procesar los paquetes capturados.

```
void setup() {
  Serial.begin(57600);  // Usa una velocidad de baudios más alta para

  // Inicializar Wi-Fi en modo estación
  WiFi.mode(WIFI_STA); // No necesita conectarse a un router
  WiFi.disconnect();   // Asegurarse de que no esté conectado a una re

  // Habilitar el modo promiscuo
  wifi_promiscuous_enable(true);

  // Configurar el callback para capturar paquetes
  wifi_set_promiscuous_rx_cb(promiscuous_callback);

  Serial.println("Modo promiscuo activado, escuchando paquetes...");
}
```

Figure 2.13: Configuración inicial del sniffer.

Se hace uso de la funcion promiscous_callback, la cual se activa cada vez que se recibe un paquete, se añade un ’0’ para los bytes hexadecimales <0x10 ya que durante las pruebas mostraron un comportamiento de eliminar el 0 a la izq, por ejemplo un valor hexadecimal x08 o x09 se mostraban como x8 o x9.

2.0.4 Modelos IA

Entre los modelos de ML (Machine Learning) que consideramos para detección de anomalías, están K-means, Isolation Forest y Random Forest.

K-means se identifica como una técnica de agrupamiento para abordar la tarea de ML de agrupamiento, que implica encontrar agrupaciones naturales de datos. En este tipo de procedimiento, no se requiere un resultado específico asociado con los datos, como ocurre en otras técnicas de ML. El objetivo principal de K-means es establecer grupos a partir de datos que no tienen una etiqueta particular pero que presentan características similares que pueden ser explotadas para conformar un

número definido de grupos. Cada grupo se define teniendo en cuenta la similitud entre los datos utilizados para realizar el análisis. El algoritmo de K-means implica asignar cada uno de los n ejemplos a uno de los k grupos, donde k es un número definido de antemano. El objetivo es minimizar las diferencias dentro de cada grupo y maximizar las diferencias entre los grupos (Muñoz and Castañeda, 2023).

Random Forest (RF) es un modelo de aprendizaje supervisado basado en un conjunto de árboles de decisión. Se utiliza tanto para clasificación como para regresión y es conocido por su robustez y precisión. RF construye múltiples árboles de decisión durante el entrenamiento y utiliza una estrategia de votación para la clasificación de datos. Esto lo vuelve resistente al sobreajuste donde su desempeño en datos entrenados es alto pero con nuevos datos de prueba o reales su desempeño decae, especialmente en conjuntos de datos grandes y complejos. Una de sus principales ventajas es su capacidad para manejar datos con características no lineales, variables categóricas y numéricas, así como para medir la importancia de las características en el modelo. Sin embargo, con grandes conjuntos de datos puede llegar a demandar muchos recursos computacionales(Matsuo et al., 2022).

Isolation Forest (IF), es un algoritmo de complejidad lineal diseñado para la detección de anomalías puntuales. Aprovecha el hecho de que las anomalías son pocas y diferentes del resto de los datos. Puede aplicarse de manera eficiente a datos multidimensionales y el paso clave del algoritmo IF es construir un conjunto de árboles de aislamiento. Un IT es una estructura de árbol binario generada aleatoriamente para aislar cada punto individual. Entre sus ventajas están que no requiere de preprocesamiento intensivo; puede manejar datos de alta dimensionalidad, es decir, funciona bien incluso con conjuntos de datos que tienen muchas características; es más rápido y escalable en comparación con otros métodos de detección de anomalías, especialmente para conjuntos de datos grandes; y no requiere etiquetas para entrenarse, lo que lo hace ideal para tareas de detección de anomalías donde las etiquetas son escasas o inexistentes (Jiang et al., 2021).

Los modelos seleccionados fueron Isolation Forest y Random Forest, el criterio para elegir IF fue su capacidad de detectar anomalías en su aprendizaje no

supervizado, lo cual puede resultar ventajoso en casos donde se desee un modelo adaptable o encontrar nuevos patrones anomalos en el trafico de red. Tambien consideramos Random Forest ya que al ser un modelo supervisado se espera que resulte más preciso ya que podemos generar ataques controlados y centrarnos en el tráfico destinado unicamente al nodo receptor.

2.1 Análisis WSN

Realizamos la visita para determinar el estado de la WSN, la red se encontraba inactiva debido a problemas relacionados con el suministro de energía tanto de los nodos como del gateway. Durante la revisión de los nodos detectamos lo siguiente:

- Uno de los nodos se encontraba algo oxidado, esto coincide con la revision inicial de la entrega de valores de humedad que distan mucho del resto de nodos.
- Uno de los nodos sensores se encontró desaparecido, sin microcontrolador ni sensor.
- Debido a las limitaciones presentadas en el periodo de cortes de luz la recopilación de datos presentó ciertas dificultades

Luego de la revision y análisis de la red actual, se instalan los materiales disponibles: 2 microcontroladores esp8266 nodeMCU y 4 sensores de humedad hd-38

A continuación, se muestra un diagrama de bloque que explicando la estrategia utilizada para desarrollar nuestro sistema de monitoreo y detección.

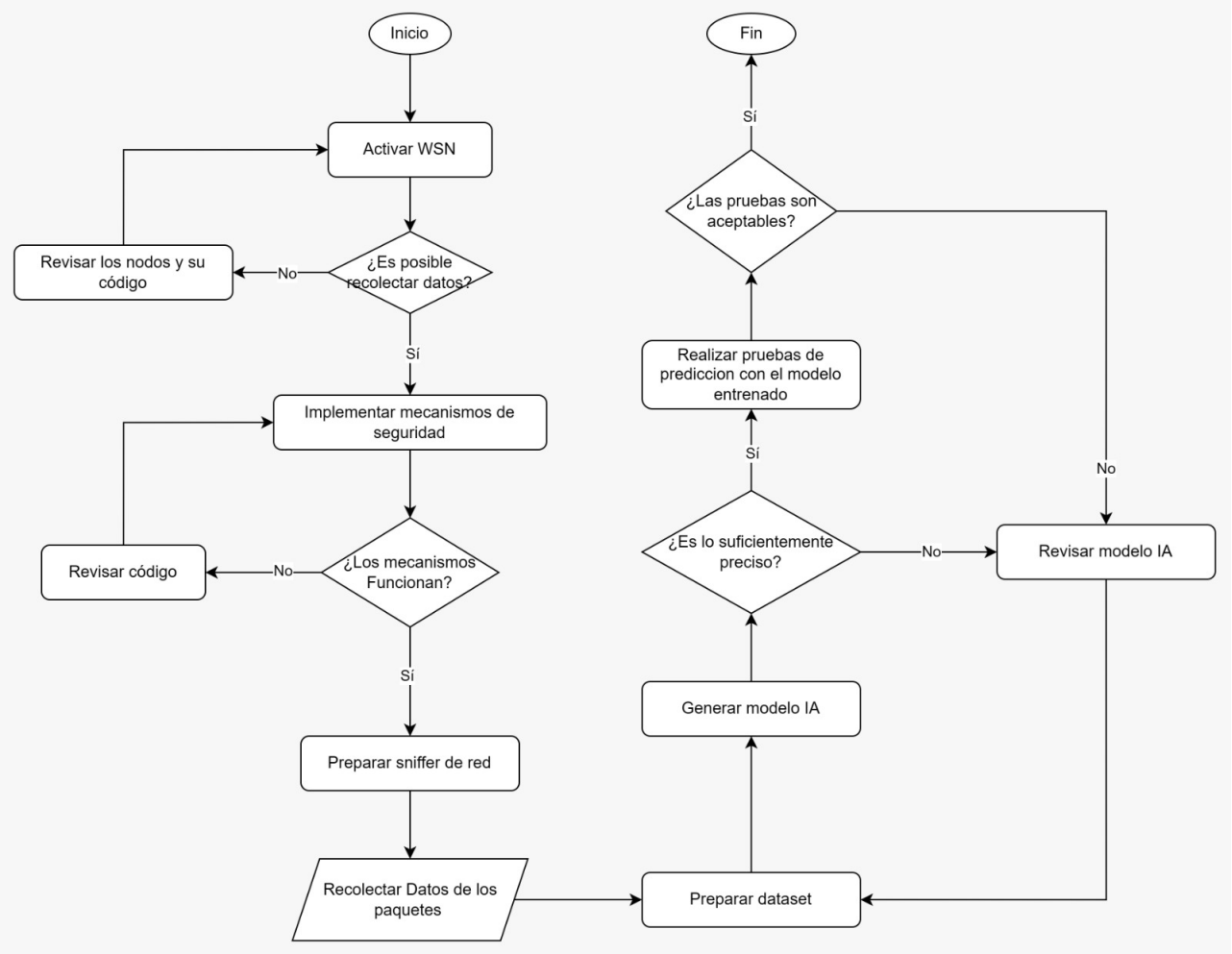


Figure 2.14: Diagrama de bloque de los métodos utilizados.

CAPÍTULO 3

3. PRUEBAS Y RESULTADOS

3.1 Pruebas

Durante la creacion de mecanismos de seguridad básicos realizamos pruebas analizando los paquetes leídos por el nodo receptor y un nodo emisor y determinar si son admitidos o rechazados. Por paquetes ”rechazados” nos referimos a los paquetes que llegaron al emisor pero no serán procesados ni enviados por el serial hacia el gateway.

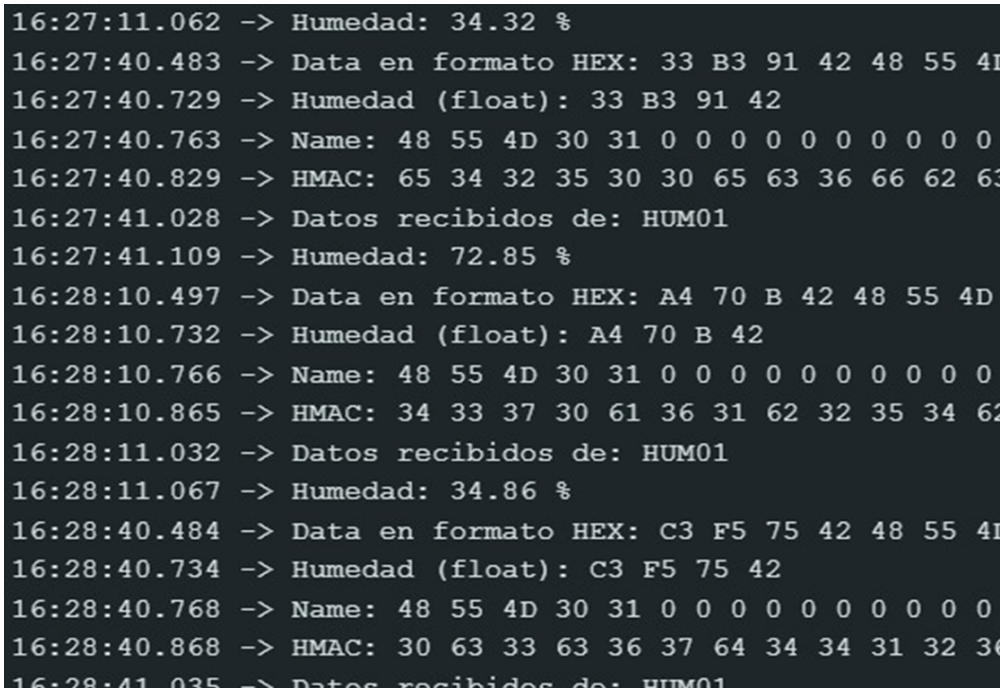


Figure 3.1: Análisis HMAC.

Como podemos observar en la figura 3.1, se muestran los valores ID y humedad en formato hexadecimal y sus valores decodificados en float y string respectivamente. Esto se realiza para comprender la posicion que ocupa el HMAC en los paquetes.

Para verificar la utilidad de los mecanismos de seguridad propuestos, AllowList

y HMAC, realizamos las siguientes pruebas:

- **AllowList**, utilizando uno de los nodos o microcontroladores disponibles y bajo el contexto que un intruso conozca la dirección MAC del receptor modificamos la lista de acceso eliminando el nodo conocido.

```
uint8_t WhiteList[][6] ={
  {0xAC, 0X0B, 0XFB, 0XD7, 0XC9, 0X02}, //Hum01
  //{0X48, 0X55, 0X19, 0X12, 0X58, 0X4C}, // //sniffer
  //{0XAC, 0X0B, 0XFB, 0XD6, 0XF0, 0XDA}, //Hum03
  {0X48, 0XE7, 0x29, 0X62, 0X70, 0X77}, // Hum02
  {0x48, 0x55, 0x19, 0x0B, 0xA2, 0xE6} //Hum05
};
```

Figure 3.2: Lista de acceso MAC.

Como podemos observar en la figura 3.2, comentamos el nodo identificado HUM03 para que el receptor rechace la comunicación entrante de esa MAC ”desconocida”. Una vez aplicada la función, el serial enviará un ”000” y nuestra Raspberry tomará esta bandera o valor como un rechazo.

- **HMAC**, para este escenario el atacante no solo tiene a disposición la direccion MAC del receptor sino que también conoce al menos una de las direcciones permitidas en la lista de acceso, sin embargo desconoce la clave que utilizan ambos dispositvos para generar el HMAC

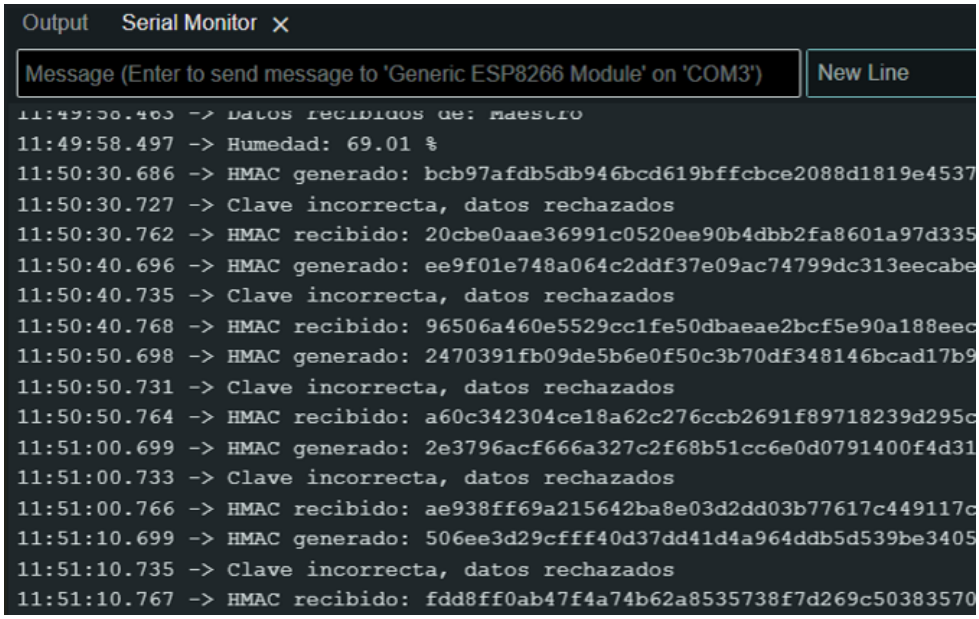


Figure 3.3: Verificar HMAC - Receptor

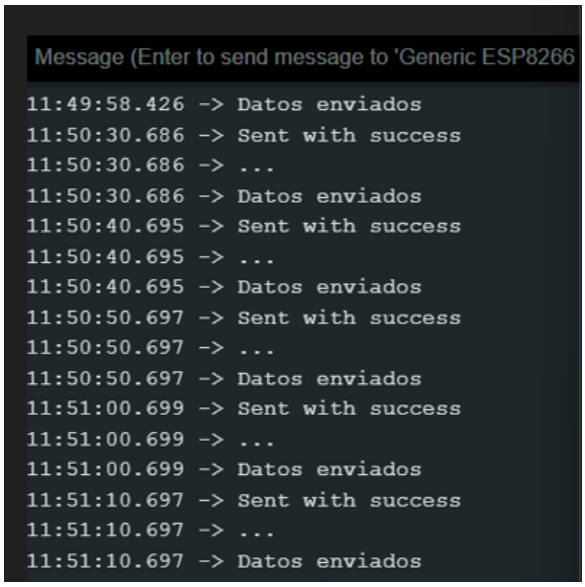


Figure 3.4: Verificar HMAC - Emisor

En las figuras 3.3 y 3.4 podemos observar el comportamiento y valores que compara la fucnion VerifyHMAC, los valores para generar el HMAC correcto y válido son los siguientes: ID, humedad, clave secreta. Estos 3 valores se utilizan para generar el HMAC, el receptor capta el ID, humedad y HMAC por separado luego procede a recrear el HMAC por su cuenta y compara el HMAC obtenido con el generado, por lo tanto si se encuentra alguna diferencia en el hmac el paquete será ”rechazado”.

Luego de verificar los mecanismos de seguridad procedemos con la búsqueda de un sniffer de red. Primero se optó por utilizar la Raspberry PI4 o gateway como sniffer ya que se encuentra dentro del área de alcance de la red, sin embargo, se presentaron obstáculos que impidieron la captura de los paquetes esp-now() con la tarjeta de red integrada, por ejemplo: al utilizar esp-now, no configuramos los microcont como punto de acceso o AP, por lo tanto no podremos ”escuchar” el tráfico.

Al no conseguir resultados con adaptadores de red conocidos, decidimos utilizar uno de los microcontrolador esp8266 disponibles y configurarlo en modo ”promiscuo” para utilizarlo como sniffer.

```
[2024-11-29 12:20:17]
xE1x0Bx57x50x00x00x00x00x00x00x01x00xD0x00x3Cx00x48x55x19x0BxA2xE6x48xE7x29x62x70x77x48x55x19x0BxA2xE6x
0xCFxE5xE0x70x0Ax58xD9x24x0BxFAxCDxEBx45xFFxB4xC7x57x10x4CxE8x49x76x05x83x37x43x94x3FxD2xB4x00x00x00x88
00x57x00
[2024-11-29 12:21:15]
xC8x0Bx57x50x00x00x00x00x00x00x01x00xD0x00x3Cx00x48x55x19x0BxA2xE6xACx0BxFBxD6xF0xDAx48x55x19x0BxA2xE6x
2x21xA3x08x41xB7xD0xE1xD3xF7x43x0Fx0Cx91xBCx14x0ExF9x9Ex39xD1x74xD2xFEx10x10x61x39x3Dx80xD2x00x00x00xA3
00x57x00
[2024-11-29 12:21:18]
xD9x0Bx57x50x00x00x00x00x00x00x01x00xD0x00x3Cx00x48x55x19x0BxA2xE6xACx0BxFBxD7xC9x02x48x55x19x0BxA2xE6x
0x3CxE7x02x22x79xC8xCBxCEx56x2Dx2DxC0xC5x17xA9x7AxC8x74xA5xB1xA8xA1xE3xAFxA8xE3xB2xA3x32x66x00x00x00xA3
00x57x00
[2024-11-29 12:21:47]
xC7x0Bx57x50x00x00x00x00x00x00x01x00xD0x00x3Cx00x48x55x19x0BxA2xE6xACx0BxFBxD6xF0xDAx48x55x19x0BxA2xE6x
2x21xA3x08x41xB7xD0xE1xD3xF7x43x0Fx0Cx91xBCx14x0ExF9x9Ex39xD1x74xD2xFEx10x10x61x39x3Dx80xD2x00x00x00xC9
00x57x00
[2024-11-29 12:21:48]
xD6x0Bx57x50x00x00x00x00x00x00x01x00xD0x00x3Cx00x48x55x19x0BxA2xE6xACx0BxFBxD7xC9x02x48x55x19x0BxA2xE6x
5x51x78x99x35x29x7BxBFx42xDCx8Cx33x4Dx4Ex01x72xD2x06x62xBFx37x18x22x44x7Ax74x9ExA7x8Bx79x13x00x00x00x4E
00x57x00
```

Figure 3.5: Paquetes capturados por el sniffer.

Los resultados fueron inmediatos, tal como vemos en la figura 3.5 logramos captar tráfico en tiempo real, todo en valores hexadecimales. Utilizando identificadores únicos para Espressif filtramos el tráfico relevante y analizamos los paquetes recibidos para conocer que significan los valores hexadecimales en los distintos segmentos de trama. Para las pruebas iniciales el filtro utilizado fue ”x18FE34”, el cual es un identificador de la organizacion Espressif. En la imagen mostrada a continuación, figura 3.6, tenemos el filtro utilizado segmentar los paquetes, este filtro se obtuvo luego de analizar minuciosamente las partes de los paquetes obtenidos.

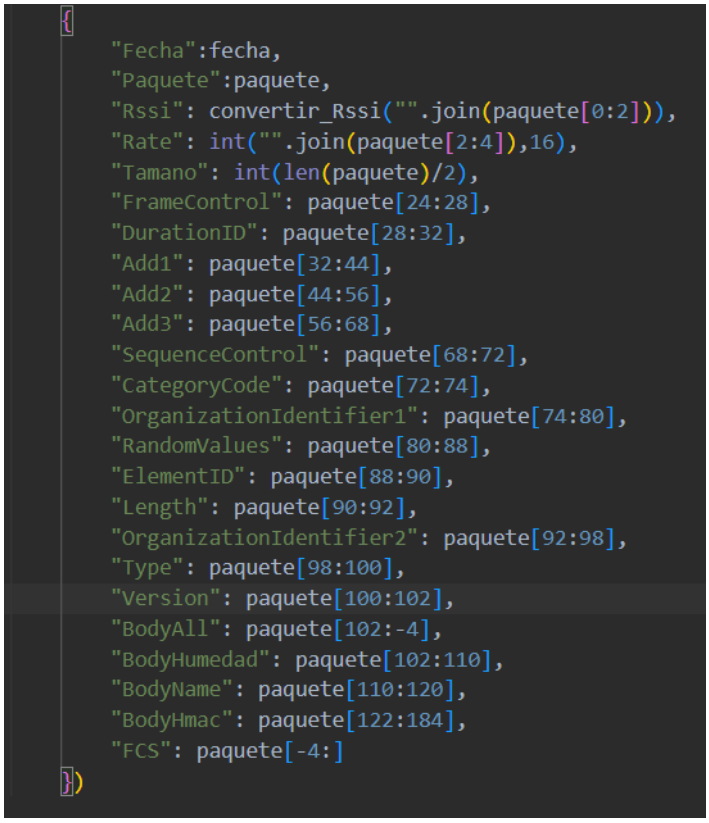
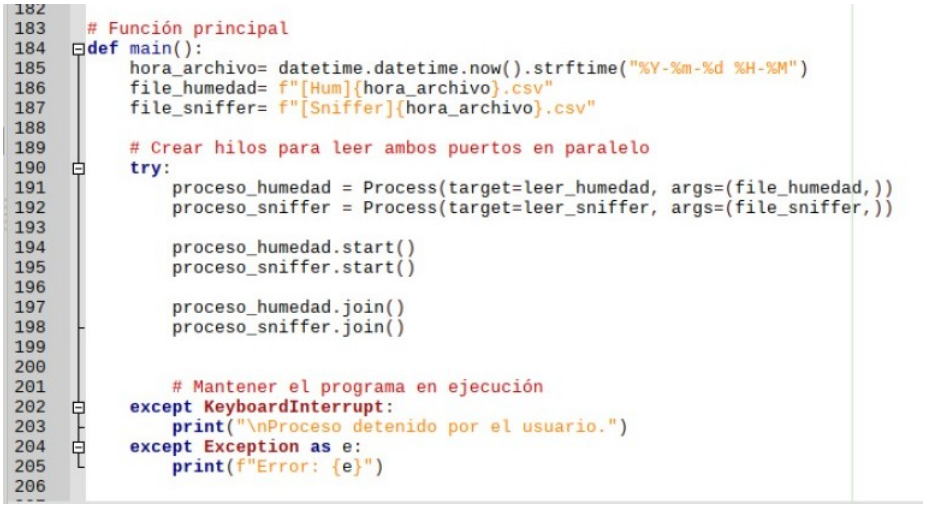


Figure 3.6: Segmentos de trama conocidos.

Los campos de interés o relevantes que utilzamos para predecir ataques tipo DoS son los siguientes:

- **DA**, Destination Address es la direccion MAC del dispositivo que recibe el paquete
- **SA**, Source Address es la direccion MAC del dispositivo que envía el mensaje
- **FrameControl**, indicador para el tipo de paquete transmitio. Data, managment o control.
- **RSSI**, es un indicador de la intensidad de señal recibida, este campo es obtenido por medio de los microcontroladores.
- **Rate**, tasa de envio de datos, definido por el estandar de WiFi utilizado.

En la figura 3.7 mostramos el uso de múltiples procesos para manejar lectura serial de sniffer y nodo receptor simultáneamente, esto presentó algunas dificultades ya que la lectura de puertos parecía interferir entre sí. Es importante manejar distintos procesos para garantizar el funcionamiento de todo el sistema, ya que utilizamos distintos puertos seriales para recibir la información.



```
182
183  # Función principal
184  def main():
185      hora_archivo= datetime.datetime.now().strftime("%Y-%m-%d %H-%M")
186      file_humedad= f"[Hum]{hora_archivo}.csv"
187      file_sniffer= f"[Sniffer]{hora_archivo}.csv"
188
189      # Crear hilos para leer ambos puertos en paralelo
190      try:
191          proceso_humedad = Process(target=leer_humedad, args=(file_humedad,))
192          proceso_sniffer = Process(target=leer_sniffer, args=(file_sniffer,))
193
194          proceso_humedad.start()
195          proceso_sniffer.start()
196
197          proceso_humedad.join()
198          proceso_sniffer.join()
199
200
201          # Mantener el programa en ejecución
202      except KeyboardInterrupt:
203          print("\nProceso detenido por el usuario.")
204      except Exception as e:
205          print(f"Error: {e}")
206
```

Figure 3.7: Multiprocesos.

Para la recopilación de datos trabajamos en 2 áreas o escenarios distintos, para el entorno de prueba lo llamaremos área A y para el de practica LST. Los datasets utilizados para entrenamiento y pruebas son los siguientes:

- Filtro consta de 3230 registros con 439 como DoS.
- 30seg consta de 8708 registros con 1000 como DoS.
- 30segCasa consta de 50367 registros con 6391 como DoS.
- Merge consta de 721 registros con 100 como DoS.
- RF-LST-Test consta de 16811 registros con 1432 como DoS

3.1.1 Pruebas A - ISOLATION FOREST

Para el entorno de prueba usamos un área residencial donde no intervienen, o al menos el sniffer no captó, otros dispositivos ESP a su alrededor. Los intervalos entre cada envío de humedad se decidió establecerlo en en 31, 32 y 35 segundos para los nodos HUM01, HUM02 y HUM05 respectivamente, esta demora es ideal para conseguir datasets mas completos.

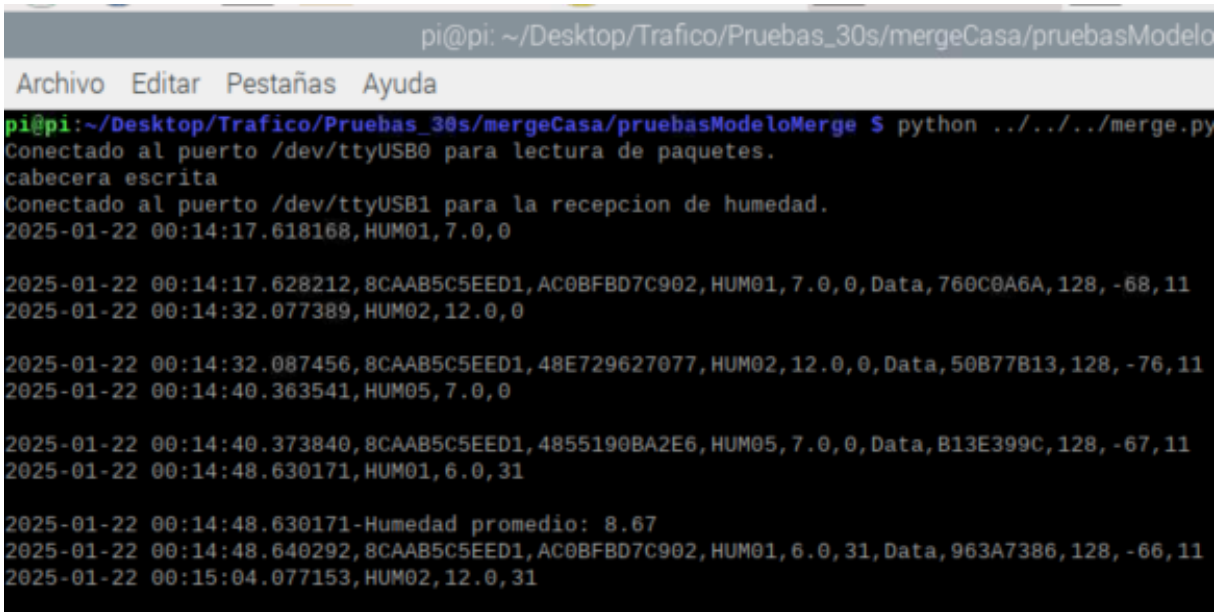


Figure 3.8: Captura de trafico y valores de humedad.

En la figura 3.8 podemos observar los valores de humedad entregados por el nodo receptor y el paquete capturado por el sniffer, con estos dos paquetes evidenciados simultaneamente hemos comprobado el monitoreo del tráfico en la WSN. También podemos apreciar el procesamiento aplicado a los paquetes obtenidos por el sniffer, se han segmentado algunas de las características previamente mencionadas Además de los datos obtenidos por el paquete tambien tenemos algunos valores agregados:

- **Fecha**, Marca temporal asignada cada vez que se lee por serial, tanto para registros del sniffer como para el receptor.
- **Emisor y Humedad**, Codificación del cuerpo del paquete, la informacion que se quiere transmitir entre un nodo sensor y el receptor. Estos campos representan la identifacion del nodo sensor que envía el paquete y el valor de humedad que desea trasmitir.
- **Intervalo**, obtenido mediante el calculo de las marcas temporales por cada SA registrada por el sniffer en el trafico de la WSN, tráfico dirigido hacia el receptor.

Para generar los ataques DoS, utilizamos el mismo nodo de prueba para los mecanismos de seguridad, enviamos los mensajes con un payload de 250 bytes con un delay 0.015, como conocemos los tiempos y el contenido del ataque DoS podemos clasificarlos luego de la generación de los archivos de recopilación .csv tal como se muestra en la figura 3.9

30	2025-01-23 15:51:23.888405	8CAAB5C5EED1	3	HUM01	6	0	Data	A24F9521	128	-86	0	Normal
31	2025-01-23 15:51:43.191230	8CAAB5C5EED1	0	HUM05	6	34	Data	081FEC24	128	-88	11	Normal
32	2025-01-23 15:51:54.887277	8CAAB5C5EED1	3	HUM01	6	30	Data	4D0731D6	128	-93	0	Normal
33	2025-01-23 15:52:15.255832	8CAAB5C5EED1	1	HUM02	13	64	Data	12FFFEAB	128	-78	11	Normal
34	2025-01-23 15:52:18.189198	8CAAB5C5EED1	0	HUM05	7	34	Data	726F067F	128	-87	11	Normal
35	2025-01-23 15:52:25.932878	8CAAB5C5EED1	3	HUM01	5	31	Data	644D1C7B	128	-90	11	Normal
36	2025-01-23 15:52:47.252496	8CAAB5C5EED1	1	HUM02	9	31	Data	BAD308D9	128	-77	11	Normal
37	2025-01-23 15:52:47.275279	8CAAB5C5EED1	1	HUM02	9	0	Data	BAD308D9	128	-80	0	Normal
38	2025-01-23 15:52:53.188910	8CAAB5C5EED1	0	HUM05	8	34	Data	272CFDB7	128	-88	11	Normal
39	2025-01-23 15:52:53.200349	8CAAB5C5EED1	0	HUM05	8	0	Data	272CFDB7	128	-87	0	Normal
40	2025-01-23 15:53:51.254769	8CAAB5C5EED1	1	HUM02	13	63	Data	69A1D7A8	128	-76	11	Normal
41	2025-01-23 15:53:58.933414	8CAAB5C5EED1	3	HUM01	7	93	Data	634B4744	128	-86	11	Normal
42	2025-01-23 15:54:07.937437	8CAAB5C5EED1	2			0	Data		128	-66	11	DoS
43	2025-01-23 15:54:07.982688	8CAAB5C5EED1	2			0	Data		128	-65	11	DoS
44	2025-01-23 15:54:08.154777	8CAAB5C5EED1	2			0	Data		128	-66	11	DoS
45	2025-01-23 15:54:08.168749	8CAAB5C5EED1	2			0	Data		128	-65	11	DoS
46	2025-01-23 15:54:08.183112	8CAAB5C5EED1	2			0	Data		128	-66	11	DoS
47	2025-01-23 15:54:08.204104	8CAAB5C5EED1	2			0	Data		128	-66	11	DoS
48	2025-01-23 15:54:08.218106	8CAAB5C5EED1	2			0	Data		128	-66	11	DoS
49	2025-01-23 15:54:08.224941	8CAAB5C5EED1	2			0	Data		128	-65	11	DoS
50	2025-01-23 15:54:08.490402	8CAAB5C5EED1	2			0	Data		128	-65	11	DoS
51	2025-01-23 15:54:08.512374	8CAAB5C5EED1	2			0	Data		128	-64	11	DoS
52	2025-01-23 15:54:08.527633	8CAAB5C5EED1	2			0	Data		128	-65	11	DoS
53	2025-01-23 15:54:08.536020	8CAAB5C5EED1	2			0	Data		128	-65	5	DoS

Figure 3.9: Clasificacion Normal o DoS

Una vez realizada la recopilación de datos tanto de datos normales como de ataques comenzamos podemos continuar con la generación de nuestro modelo estableciendo algunos parametros:

- **Entrenamiento**, Porcentaje del dataset destinado para el entrenamiento del modelo, utilizamos un 80% de la totalidad del dataset.
- **Pruebas inmediatas**, Porcentaje del dataset destinado para pruebas dentro del mismo dataset, utilizamos el 20% restante.
- **Contaminacion**, se utiliza en IsolationForest y es el valor que indica el porcentaje de datos anomalos esperados.

Inicialmente decidimos utilizar el modelo IsolationForest que es un modelo no supervisado para detectar anomalias. Para esta prueba utilizamos el dataset 30seg.xlsx, en la figura 3.10 podemos ver una matriz de confusión utilizada para verificar que las predicciones realizadas por el modelo son correctas o no, se puede apreciar que la mayor cantidad de data representa registros normales, esto se debe a que es necesario identificar las anomalias como comportamiento extraño, para poder diferenciar con el algoritmo los valores que distan mucho del resto.

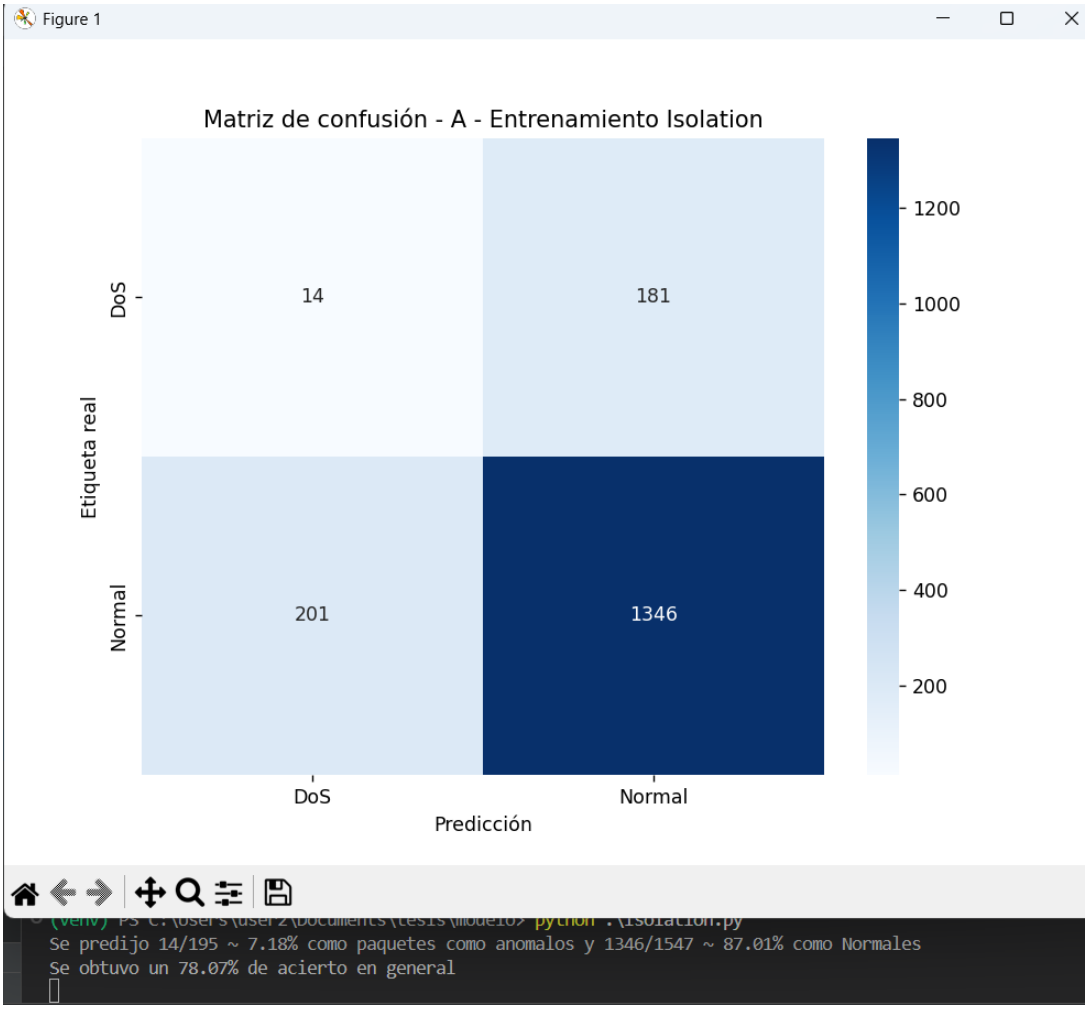


Figure 3.10: Modelo Isolation Forest prueba.

El modelo tiene un porcentaje de acierto general del 78.07%, sin embargo si tomamos en cuenta el porcentaje de acierto para datos DoS el porcentaje es de acierto es de apenas 7.18%. Se continuaron realizando pruebas con otros datasets o campos de interés pero los resultados no mejaron mucho por lo tanto se decidió utilizar el modelo RandomForest para continuar con las pruebas. No se realizaron pruebas en vivo para este modelo.

3.1.2 Pruebas A - RANDOM FOREST

Las condiciones para las pruebas con Random Forest son las mismas mencionadas en Pruebas A - ISOLATION FOREST, con la diferencia que no tenemos que indicar el porcentaje de ”Contaminacion” o registros anómalos debido a que podemos utilizar directamente la clasificación Normal o DoS que realizamos Para esta

prueba se utilizó el dataset 30segCasa, de igual manera se usará el 80% de este dataset para entrenamiento y el 20% para pruebas dentro de la misma iteración.

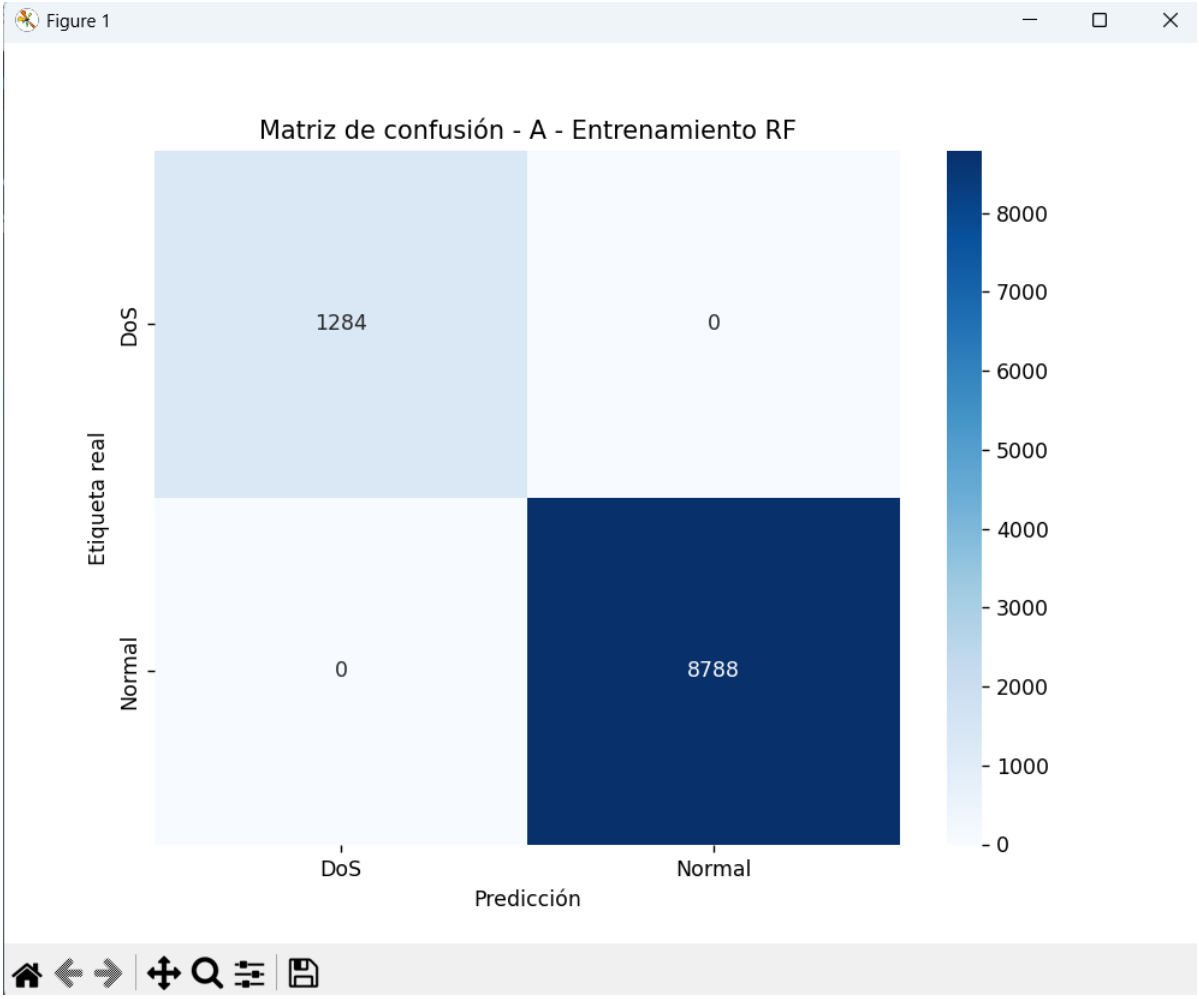


Figure 3.11: Modelo Random Forest prueba entrenamiento.

En la figura 3.11 podemos ver los resultados del entrenamiento, se han predicho con un acierto del 100% los registros Normales y DoS. Al tener un modelo capaz de detectar correctamente los ataques DoS que hemos definido procedemos a realizar pruebas en vivo, para estas pruebas utilizaremos el codigo de recoleccion de datos y mostraremos por serial los resultados del modelo. El código trabaja por lotes de 6 registros, sean 6 registros normales que tomarían en promedio 1 minuto en enviarse debido al intervalo de 30 segundos configurado, o en el caso de un ataque poder captar y predecir el mismo de manera inmediata. En la figura 3.12 mostrada tenemos el microcontrolador atacante, se encuentra muy cerca de toda la red, recordemos que estamos en las pruebas A un ambiente controlado.

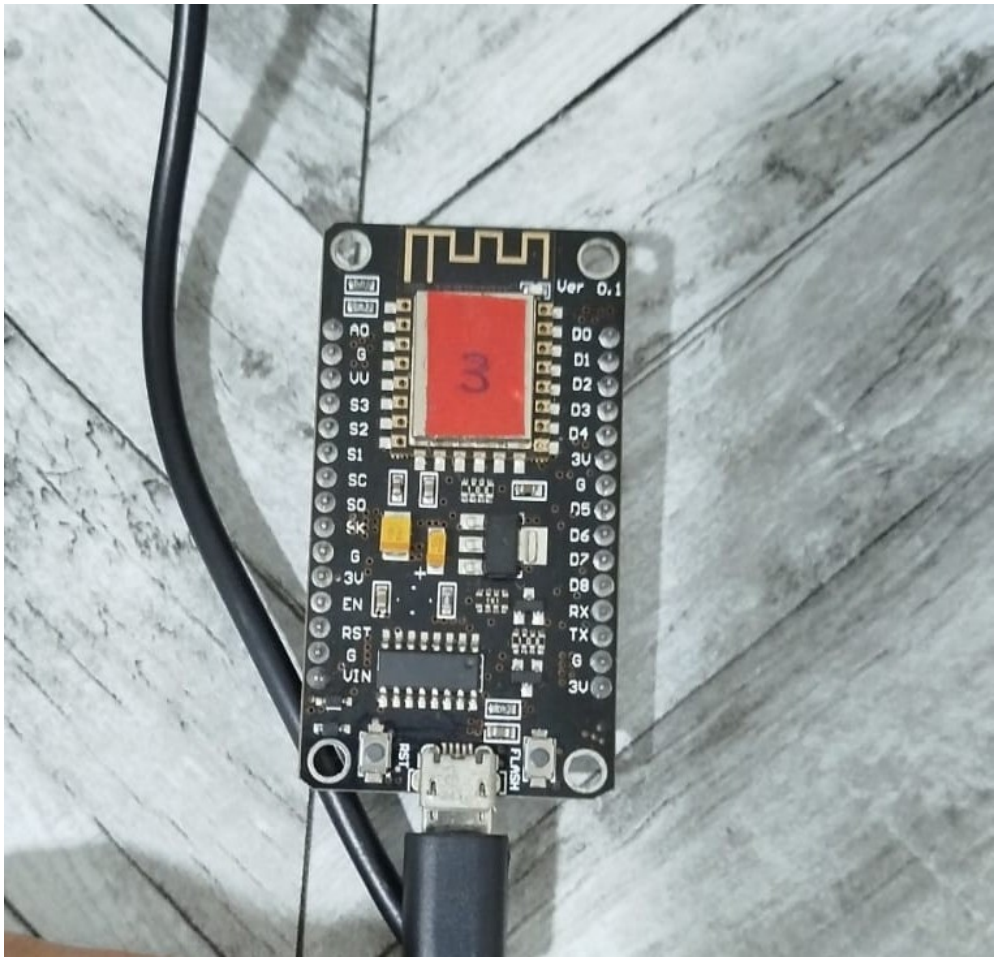


Figure 3.12: Nodo atacante

Una vez comprendido como efectuar los ataques DoS y teniendo un modelo listo para usar, procedemos a realizar las pruebas en directo para el área A, esta prueba la realizamos con un dataFrame de apoyo para el modelo, donde el contenido es trafico permitido o valido dentro de la WSN para que sea más preciso.

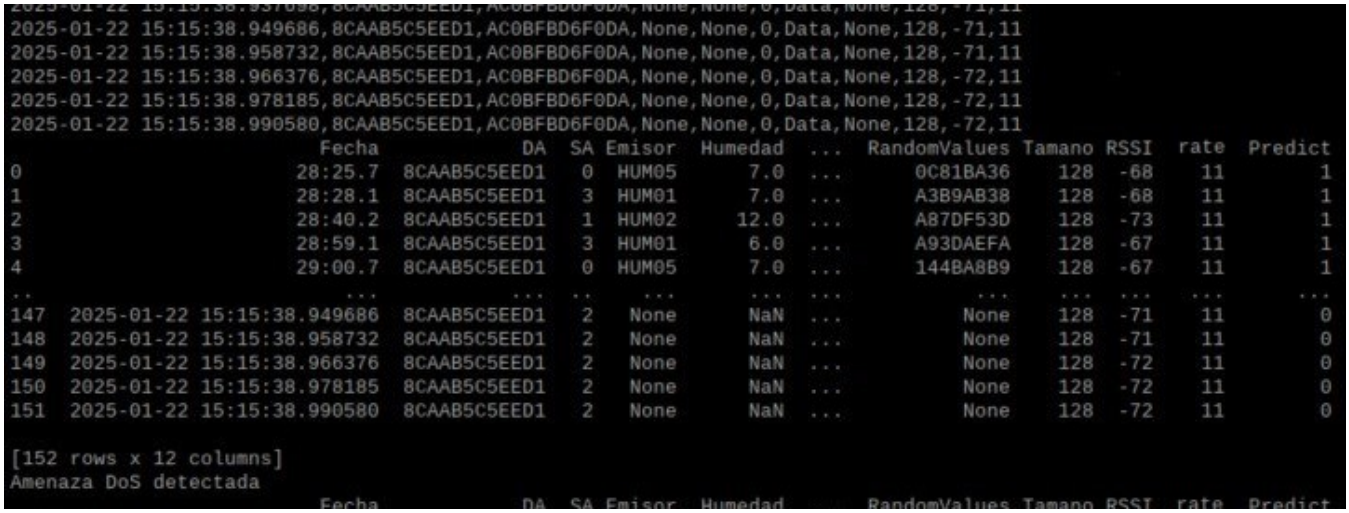


Figure 3.13: Modelo RF predict pruebas

En la figura 3.13 observamos el dataframe con la columna predict, la funcion predict por parte del modelo RandomForest entrega valores 0 o 1, anomalo o normal respectivamente. Con estos valores representando tráfico normal o tráfico anómalo podemos comparar la cantidad de valores 0 obtenidos y determinar si se está siendo blanco de un ataque DoS o en caso contrario calcular el valor de 1 obtenidos y determinar que no hay afectaciones en el tráfico de la WSN.

3.1.3 Pruebas LST - Random Forest

Estas pruebas se realizaron luego de completarse las pruebas en el entorno controlado pruebas A y únicmamente se trabajó con Random Forest ya que Isolation Forest no entregaba resultados favorables en las pruebas. Alrededor del LST existen otros proyectos que hacen uso de dispositivos esp8266, esto lo evidenciamos en la etapa de análisis ya que lograbamos capturar paquetes de dispositivos no conocidos y como nuestro filtro estaba basado en el identificador Espressif logramos capturar paquetes tipo Data, Control o Managment por ejemplo nodos en modo Acces Point anunciando su red. Luego de este análisis cambiamos el filtro para analizar únicamente el tráfico destinado a nuestro receptor.

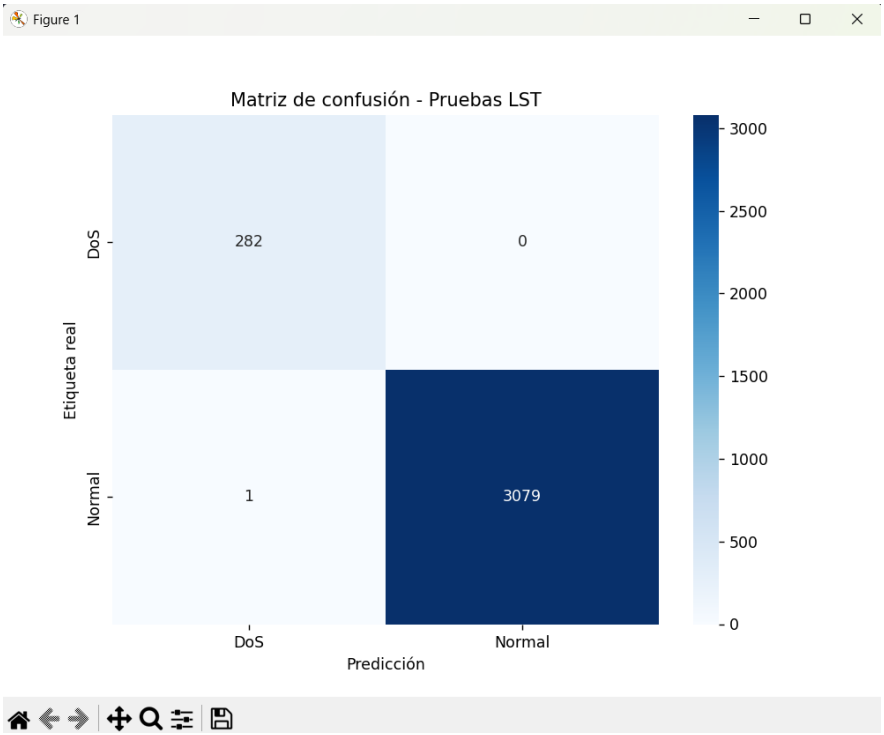


Figure 3.14: LST-RF entrenamiento

En la figura 3.14 podemos ver la matriz de confusión obtenida de las pruebas luego del entrenamiento con el dataset RF-LST-Test que fue recopilada en el LST, hay que recordar que las condiciones son distintas, la distancia entre los nodos y el receptor, la interferencia presente por el resto de redes, el ingreso de estudiantes con proyectos utilizando ESP, esto se menciona para denotar la importancia de generar un nuevo dataset que represente los valores reales de transmision, intervalos, o interferencias. A continuación mostraremos capturas acerca del uso del modelo para la WSN activa.

3.2 Resultados

3.2.1 Métricas de Evaluación

Para determinar el desempeño de nuestro modelo se emplearon varias métricas calculadas a partir de la obtención y clasificación de resultados obtenidos. Estas métricas son las mismas que utiliza la matriz de confusión generada en python.:

- **TP (True Positives):** Representa la cantidad de casos que fueron correctamente

clasificados como positivos, es decir, cuando el modelo identificó correctamente los ataques DoS.

- **TN (True Negatives):** Indica los casos correctamente clasificados como negativos, lo que corresponde al tráfico normal.
- **FP (False Positives):** Refleja los casos clasificados incorrectamente como positivos
- **FN (False Negatives):** Se refiere a los casos que no fueron detectados como positivos cuando en realidad lo eran, clasificar normal cuando en realidad eran DoS

Con estos valores podemos calcular la exactitud, precisión, y muchas otras métricas para determinar la efectividad del modelo.

Exactitud (Accuracy): Esta métrica mide la proporción total de predicciones correctas realizadas por el modelo en relación con el número total de predicciones. Representa la capacidad general del modelo para clasificar correctamente tanto los casos positivos como negativos:

$$\text{Exactitud} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

Un valor alto de exactitud indica un buen desempeño global del modelo.

Precisión (Precision): La precisión se centra en la calidad de las predicciones positivas realizadas por el modelo. Calcula la proporción de casos clasificados como positivos que realmente lo son:

$$\text{Precisión} = \frac{TP}{TP + FP} \tag{3.2}$$

Una alta precisión asegura que el modelo minimice las falsas alarmas y confiar plenamente en las predicciones que realiza.

Sensibilidad (Recall) o Tasa de Verdaderos Positivos: Esta métrica mide la capacidad del modelo para identificar correctamente todos los casos positivos reales dentro del

conjunto de datos. Es especialmente útil en situaciones donde la detección de ataques es crítica:

$$\text{Sensibilidad} = \frac{TP}{TP+FN} \tag{3.3}$$

Un valor elevado de sensibilidad refleja la habilidad del modelo para detectar ataques sin omisiones significativas.

Especificidad: La especificidad evalúa la proporción de casos negativos correctamente identificados:

$$\text{Especificidad} = \frac{TN}{TN+FP} \tag{3.4}$$

Es una métrica clave para medir la capacidad del modelo de evitar clasificar erroneamente tráfico normal como DoS.

F1-Score: El F1-Score combina precisión y sensibilidad en una sola métrica, calculando su media armónica. Es especialmente útil cuando hay un desbalance entre clases, ya que equilibra ambos aspectos del desempeño del modelo:

$$F1 = 2 \cdot \frac{\text{Precisión} \cdot \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} \tag{3.5}$$

Un F1-Score alto indica que el modelo tiene un rendimiento equilibrado, reduciendo tanto los falsos positivos como los falsos negativos.

3.2.2 A - Random Forest

Se utilizaron distintos datasets para validar el modelo. Los dataset denominados merge y 30seg fueron recopilados bajo las mismas condiciones que el dataset utilizado para entrenamiento 30segCasa, mientras que filtro es un dataset recopilado en la etapa de analisis y pruebas con un intervalo de 300 segundos o 5 minutos por cada envío de paquetes, sin embargo los resultados fueron igual de certeros.

En la figura 3.15 se puede evidenciar que el dataset utilizado para el entrenamiento fue adecuado ya que se obtuvo un 100% de acierto general,

otorgandonos así la confianza para validar el modelo con el resto de datasets mencionados. Este será el dataset que genera al modelo ya que es el que contiene la mayor cantidad de datos recopilados.

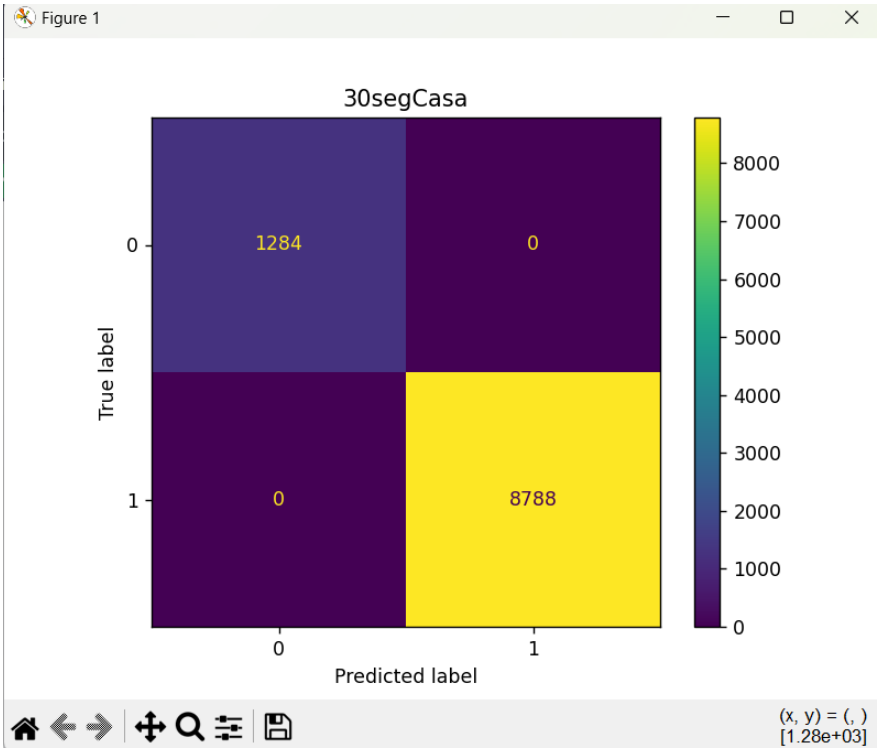


Figure 3.15: Matriz de confusión RF.

Para esta ronda de entrenamiento obtenemos las siguientes métricas de evaluación.

Valores: $TN = 8788$, $TP = 1284$, $FP = 0$, $FN = 0$

$$\text{Exactitud} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{1284+8788}{1284+8788+0+0} = 1.0 \tag{3.6}$$

$$\text{Precisión} = \frac{TP}{TP+FP} = \frac{1284}{1284+0} = 1.0 \tag{3.7}$$

$$\text{Sensibilidad} = \frac{TP}{TP+FN} = \frac{1284}{1284+0} = 1.0 \tag{3.8}$$

$$\text{Especificidad} = \frac{TN}{TN+FP} = \frac{8788}{8788+0} = 1.0 \tag{3.9}$$

$$F1 = 2 \cdot \frac{\text{Precisión} \cdot \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} = 2 \cdot \frac{1.0 \cdot 1.0}{1.0 + 1.0} = 1.0 \tag{3.10}$$

Con estas métricas podemos asegurar que el conjunto de datos de entrenamiento es adecuado para predecir el tráfico en la red y lo que sigue son las pruebas con el modelo generado. Se carga el modelo y se ingresan los los registros a ser predecidos.

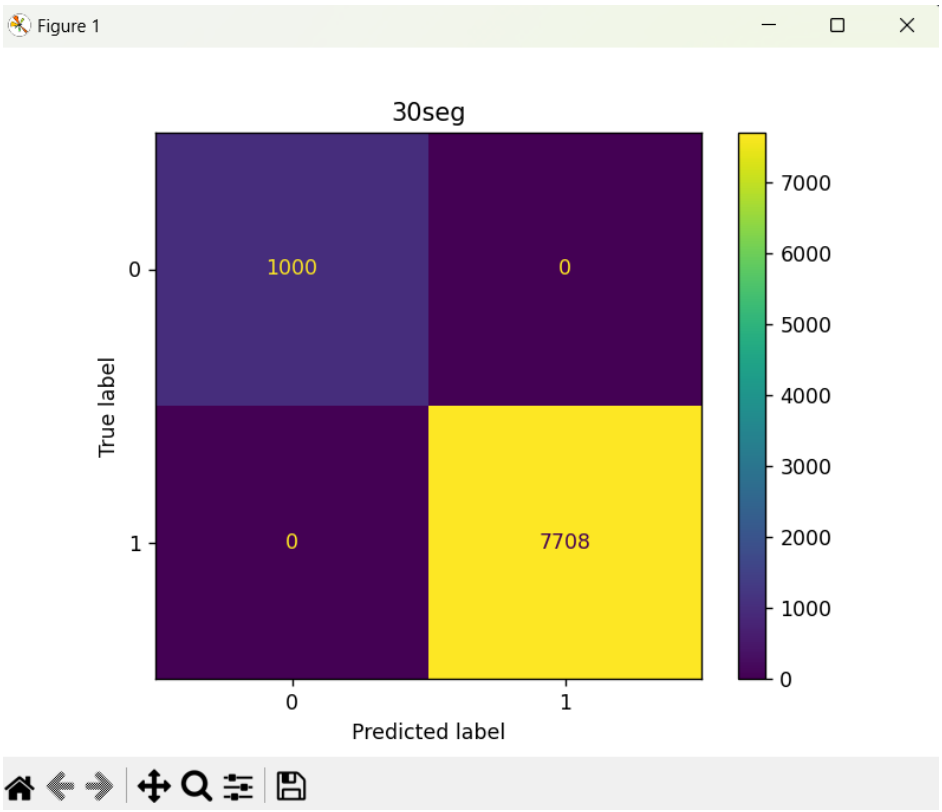


Figure 3.16: Matriz de confusión RF - 30seg.

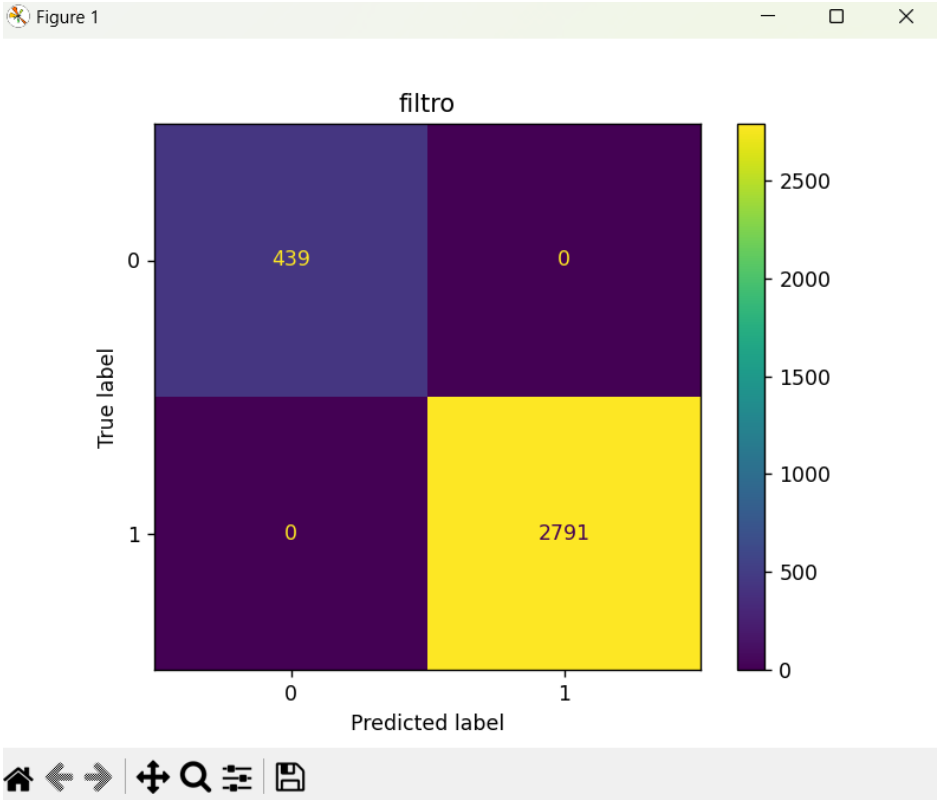


Figure 3.17: Matriz de confusión RF - filtro.

Con un vistazo rápido a las figuras 3.16 y 3.17 podemos observar que los resultados de Falsos positivos o Falsos negativos son 0 y por lo tanto los resultados de las métricas serán iguales a los calculados para el periodo de entrenamiento. Esto refleja la exactitud del modelo al predecir correctamente otros conjunto de datos y pruebas en vivo, además nos indica que el proceso de selección de características fue efectivo, identificando parámetros relevantes como: Direccion de destino, Intervalo entre paquetes, intensidad de señal recibida y tasa de envío de paquetes es posible predecir que paquetes esp-now son producto de un trafico anómalo o DoS.

3.2.3 LST - Random Forest

Para las pruebas finales realizadas en el LST, se recopiló un nuevo conjunto de datos (RF-LST-Test), logrando un porcentaje de acierto del 99.9%. Podemos observar los resultados en la martriz de confusión presentada en la figura 3.14 o también en las métricas de evaluación presentadas a continuación: Valores: $TN = 3079$, $TP = 282$, $FP = 0$, $FN = 1$

Valores: $TN = 3079$, $TP = 282$, $FP = 1$, $FN = 1$

$$\text{Exactitud} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{282+3079}{282+3079+1+1} = 0.9994 \tag{3.11}$$

$$\text{Precisión} = \frac{TP}{TP+FP} = \frac{282}{282+1} = 0.9965 \tag{3.12}$$

$$\text{Sensibilidad} = \frac{TP}{TP+FN} = \frac{282}{282+1} = 0.9965 \tag{3.13}$$

$$\text{Especificidad} = \frac{TN}{TN+FP} = \frac{3079}{3079+1} = 0.9997 \tag{3.14}$$

$$F1 = 2 \cdot \frac{\text{Precisión} \cdot \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} = 2 \cdot \frac{0.9965 \cdot 0.9965}{0.9965 + 0.9965} = 0.9965 \tag{3.15}$$

Hubo un falso negativo, el modelo predijo incorrectamente un registro como DoS cuando en realidad era tráfico normal, esto baja la precición del modelo a un 99.96% pero aún se considera un resultado muy favorable. Esta disminución en la precisión se encontraba dentro de lo esperado ya que trabajamos en un ambiente real donde las condiciones de la red pueden cambiar, donde existen otros dispositivos de comunicacion WiFi alrededor, incluso otros microcontroladores de la familia ESP.

En las figuras 3.18 y 3.19 tenemos las pruebas en vivo del modelo, tanto para un tráfico normal y para los ataques simulados. Con estas pruebas finales en el entorno del LST podemos presentar los acertados resultados entregados por nuestro modelo.

```
2025-01-23 15:48:31.247048,HUM02,10.0,32

2025-01-23 15:48:31.264602,8CAAB5C5EED1,48E729627077,HUM02,10.0,31,Data,6EF46B88,128,-80,11
2025-01-23 15:48:48.189633,HUM05,7.0,35

2025-01-23 15:48:48.800972,HUM01,6.0,31

2025-01-23 15:48:48.812759,8CAAB5C5EED1,AC0BFBD7C902,HUM01,6.0,30,Data,483081E8,128,-89,11
                        Fecha            DA  SA Emisor  Humedad  ...  RandomValues Tamano RSSI Canal  Predict
0                     28:25.7  8CAAB5C5EED1   0  HUM05      7.0  ...      0C81BA36    128  -68    11        1
1                     28:28.1  8CAAB5C5EED1   3  HUM01      7.0  ...      A3B9AB38    128  -68    11        1
2                     28:40.2  8CAAB5C5EED1   1  HUM02     12.0  ...      A87DF53D    128  -73    11        1
3                     28:59.1  8CAAB5C5EED1   3  HUM01      6.0  ...      A93DAEFA    128  -67    11        1
4                     29:00.7  8CAAB5C5EED1   0  HUM05      7.0  ...      144BA8B9    128  -67    11        1
..                        ...           ...  ..    ...      ...  ...           ...    ...  ...   ...      ...
147  2025-01-23 15:48:13.197480  8CAAB5C5EED1   0  HUM05      7.0  ...      2A7E098C    128  -88    11        1
148  2025-01-23 15:48:13.208725  8CAAB5C5EED1   0  HUM05      7.0  ...      2A7E098C    128  -90    11        1
149  2025-01-23 15:48:17.911630  8CAAB5C5EED1   3  HUM01      7.0  ...      111C97AB    128  -89     5        1
150  2025-01-23 15:48:31.264602  8CAAB5C5EED1   1  HUM02     10.0  ...      6EF46B88    128  -80    11        1
151  2025-01-23 15:48:48.812759  8CAAB5C5EED1   3  HUM01      6.0  ...      483081E8    128  -89    11        1

[152 rows x 12 columns]
Todo normal
2025-01-23 15:49:03.243802,HUM02,10.0,31

2025-01-23 15:49:03.243802-Humedad promedio: 7.67
2025-01-23 15:49:03.254085,8CAAB5C5EED1,48E729627077,HUM02,10.0,31,Data,029F4DD9,128,-78,11
2025-01-23 15:49:19.813249,HUM01,6.0,31
```

Figure 3.18: RF - Espol - Predict 1

```
[152 rows x 12 columns]
2025-01-23 15:54:11.768832,8CAAB5C5EED1,AC0BFBD6F0DA,None,None,0,Data,None,128,-65,11
2025-01-23 15:54:11.776816,8CAAB5C5EED1,AC0BFBD6F0DA,None,None,0,Data,None,128,-66,11
2025-01-23 15:54:11.793343,8CAAB5C5EED1,AC0BFBD6F0DA,None,None,0,Data,None,128,-65,11
2025-01-23 15:54:11.812847,8CAAB5C5EED1,AC0BFBD6F0DA,None,None,0,Data,None,128,-66,11
2025-01-23 15:54:11.820821,8CAAB5C5EED1,AC0BFBD6F0DA,None,None,0,Data,None,128,-66,11
2025-01-23 15:54:11.828784,8CAAB5C5EED1,AC0BFBD6F0DA,None,None,0,Data,None,128,-68,11
                        Fecha            DA  SA Emisor  Humedad  ...  RandomValues Tamano RSSI Canal  Predict
0                     28:25.7  8CAAB5C5EED1   0  HUM05      7.0  ...      0C81BA36    128  -68    11        1
1                     28:28.1  8CAAB5C5EED1   3  HUM01      7.0  ...      A3B9AB38    128  -68    11        1
2                     28:40.2  8CAAB5C5EED1   1  HUM02     12.0  ...      A87DF53D    128  -73    11        1
3                     28:59.1  8CAAB5C5EED1   3  HUM01      6.0  ...      A93DAEFA    128  -67    11        1
4                     29:00.7  8CAAB5C5EED1   0  HUM05      7.0  ...      144BA8B9    128  -67    11        1
..                        ...           ...  ..    ...      ...  ...           ...    ...  ...   ...      ...
147  2025-01-23 15:54:11.776816  8CAAB5C5EED1   2   None      NaN  ...          None    128  -66    11        0
148  2025-01-23 15:54:11.793343  8CAAB5C5EED1   2   None      NaN  ...          None    128  -65    11        0
149  2025-01-23 15:54:11.812847  8CAAB5C5EED1   2   None      NaN  ...          None    128  -66    11        0
150  2025-01-23 15:54:11.820821  8CAAB5C5EED1   2   None      NaN  ...          None    128  -66    11        0
151  2025-01-23 15:54:11.828784  8CAAB5C5EED1   2   None      NaN  ...          None    128  -68    11        0

[152 rows x 12 columns]
Amenaza DoS detectada
                        Fecha            DA  SA Emisor  Humedad  ...  RandomValues Tamano RSSI Canal  Predict
```

Figure 3.19: RF - Espol - Predict 0

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- La implementación de un sistema de monitoreo y detección de ataques de red es un recurso de suma importancia para prevenir diversas situaciones que pondrían en peligro la calidad o integridad de los cultivos monitoreados por una WSN. Durante la creación de este sistema hemos notado lo vulnerables que son este tipo de redes y la importancia de tener una data confiable para su posterior uso.
- Luego de analizar la WSN existente en el LST se desarrolló un sistema de seguridad que, mediante inteligencia artificial y código python, detecta ataques DoS e inyección de datos destinados al receptor de la WSN.
- Los ataques de inyección de datos fueron mitigados al bloquear estos mensajes, hasta cierto punto, mientras que, por el lado de monitoreo y deteccion DoS la efectividad o precisión del sistema en un entorno controlado (A) y real (LST) fue del 99.9% con lo cual podemos hacer uso del sistema y esperar que la data recibida sea confiable o no recibirla en caso de que sea información fraudulente, incluso recibir alertas en caso de ataques DoS y tomar las medidas pertinentes para el caso.

4.2 Recomendaciones

Se recomienda incluir banderas o tomar en cuenta el comportamiento de los nodos sensores, por ejemplo si la humedad en este caso de un nodo x es muy distinta a la del resto de nodos podría ser un indicador del malfuncionamiento del nodo. Es importante tomar en consideración que existen más ataques que se pueden realizar a la WSN y estar preparado ante ello, por lo tanto es recomendable realizar un análisis de

vulnerabilidades y determinar cuales pueden ser corregidas, mitigadas o controladas y actuar en consecuencia. Mejorar el sistema de abastecimiento de energía o incluir un respaldo de energía.

4.3 Líneas Futuras

Es posible analizar más secciones de los paquetes transmitidos por la red e incluso utilizar información otorgada por RxControl en el sniffer, es posible identificar otros tipos de ataque como Relay, el cual toma paquetes validos y los reenvía ya sea modificando o no su contenido, para este caso podemos identificar los campos Random Values el cual es utilizado específicamente para este motivo. El campo Random Values consta de 4 bytes con valores aleatorios, si este campo se repite varias veces es un indicador de que alguien esá reenviado paquetes validos.

CAPÍTULO 5

5. ANEXOS

5.1 Tablas de Costos

Los costos presentados a continuación representan los materiales y personal empleado en la culminación del presente proyecto durante el periodo académico 2024-2.

5.1.1 Costo de Mano de Obra

Cargo	**Salario mensual (USD)**
Ing. Telemático	885
Ing. Telemático	885
Total x 4 meses	**$7,080**

Table 5.1: Costo de mano de obra

5.1.2 Costo de Componentes Utilizados

Elemento	Precio (USD)
Sensor HD-38 x4	18
ESP8266 NodeMCU x2	17
Total	**$35**

Table 5.2: Costo de componentes utilizados

La tabla 5.1 representa el salario invertido en el personal capaz de realizar la instalación de los nuevos nodos, levantamiento de información previa al análisis, investigación acerca del protocolo de comunicación empleado en la WSN, desarrollo del software y entrega final del mismo. Mientras que la tabla 5.2 presenta los costos de la adición de nuevos nodos sensores a la red. Los costos de procesamiento se omitieron ya que la WSN actualmente cuenta con un equipo de procesamiento Raspberry PI 4 y se utilizaron programas gratuitos para el monitoreo, gestion y mantenimiento del sistema.

5.2 Códigos

5.2.1 Sniffer

```
//codigo sniffer
#include <ESP8266WiFi.h>

//estructuras wifi
struct RxControl {
  signed rssi : 8;    // Señal RSSI
  unsigned rate : 4;
  unsigned is_group : 1;
  unsigned : 1;
```

```
  unsigned sig_mode : 2;
  unsigned legacy_length : 12;
  unsigned damatch0 : 1;
  unsigned damatch1 : 1;
  unsigned bssidmatch0 : 1;
  unsigned bssidmatch1 : 1;
  unsigned MCS : 7;
  unsigned CWB : 1;
  unsigned HT_length : 16;
  unsigned SNR : 8;
  unsigned noise_floor : 8;
  unsigned ampdu_cnt : 8;
  unsigned channel : 4;
  unsigned secondary_channel : 4;
  unsigned aggregation : 1;
  unsigned stbc : 2;
  unsigned fec_coding : 1;
  unsigned sgi : 1;
  unsigned rxend_state : 8;
  unsigned ampdu_reference : 8;
  unsigned lna_all : 3;
};

// Estructura de paquete
struct sniffer_buf {
  struct RxControl rx_ctrl; // Información de control del hardware
  uint8_t buf[112];      // Datos del paquete capturado
  uint16_t cnt;          // Contador de paquetes
  uint16_t len;          // Longitud del paquete
};

//#define MAX_PACKET_SIZE 550 // Definir un tamaño máximo de paquete

// Búfer global para capturar los paquetes
//uint8_t packet_buffer[MAX_PACKET_SIZE];
//char prueba[350];
//String paquete;
```

```
void setup() {
  Serial.begin(230400);
  //mac sniffer
  //uint8_t snfMAC[]= {0x48,0x55,0x19,0x12,0x58,0x4c};

  // Inicializar Wi-Fi en modo estación
  WiFi.mode(WIFI_STA); // No necesita conectarse a un router
  //wifi_set_macaddr(STATION_IF, snfMAC);
  WiFi.disconnect(); // Asegurarse de que no esté conectado a una red

  // Habilitar el modo promiscuo
  wifi_promiscuous_enable(true);

  // Configurar el callback para capturar paquetes

  wifi_set_promiscuous_rx_cb(promiscuous_callback);

  //Serial.println("Modo promiscuo activado, escuchando paquetes...");
}
void loop() {
  // Aquí no se necesitan operaciones activas, el tráfico lo maneja el callback
}

// Callback que se ejecutará cuando se capture un paquete
void promiscuous_callback(uint8_t *buf, uint16_t len) {
  const struct sniffer_buf *sniffer = (struct sniffer_buf *)buf;

  int rssi= sniffer->rx_ctrl.rssi;
  int canal= wifi_get_channel();
  int rate = sniffer->rx_ctrl.rate;

  Serial.print(rssi);
  Serial.print(",");
  Serial.print(canal);
  Serial.print(",");
  Serial.print(rate);
  Serial.print(',');
```

```
  for (int i = 0; i < len; i++) {

    if(buf[i]< 0x10){ //cambiar por el buffer global
      Serial.print("0");
      Serial.print(buf[i], HEX);
      }
    else{
        Serial.print(buf[i], HEX);
      }

      //Serial.print(" ");
  }
  Serial.println();

}
```

5.2.2 Nodo Emisor

```
    #include <ESP8266WiFi.h>
#include <espnow.h>
#include <Crypto.h>
#include <SHA256.h>

const int sensorPin= A0;

// REPLACE WITH YOUR RECEIVER MAC Address
uint8_t SlaveAdd[] = { 0x8C, 0xAA, 0xB5, 0xC5, 0xEE, 0xD1 }; //8CAAB5C5EED1
const char *SECRET_KEY = "n3IoTiPensoL9";

typedef struct Data {
  float humedad;
  char Name[6];
  char hmac[32];
} Data;
```

```
//uint8_t payload[250];

Data data;

void generateHMAC(const char *message, char *outputHMAC) {
  SHA256 sha256;
  sha256.reset();
  sha256.update((const uint8_t*)SECRET_KEY, strlen(SECRET_KEY));
  sha256.update((const uint8_t*)message, 10);

  //uint8_t hash[32];
  sha256.finalize(outputHMAC, 32);
}

void generarHumedad(){
  strcpy(data.Name, "HUM01");
  data.humedad = analogRead(sensorPin);
  char message[10];
  memcpy(message, data.Name, 6);
  memcpy(message +6, &data.humedad, 4);
  generateHMAC(message, data.hmac);
}
void OnDataSent(uint8_t *mac_addr, uint8_t status) {
  Serial.println(status == 0 ? "Datos enviados" : "Error en el envío");
}

void setup() {
  // Init Serial Monitor
  delay(1000);
  Serial.begin(9600);
  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Initilize ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
```

```
    return;
  } else {
    Serial.println("Initialized ESP-NOW");
  }
  esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);
  esp_now_register_send_cb(OnDataSent);

  // Add peer
  if (esp_now_add_peer(SlaveAdd, ESP_NOW_ROLE_SLAVE, 0, NULL, 0) != 0) {
    Serial.println("Failed to add peer");
    return;
  } else {
    Serial.println("Added peer");
  }

  if (esp_now_is_peer_exist(SlaveAdd)) {
    Serial.println("Peer exists");
  } else
    Serial.println("No exists");
}

void loop() {
  delay(10);
  Serial.print(data.humedad);
  generarHumedad();
  int result = esp_now_send(SlaveAdd, (uint8_t *)&data, sizeof(data));
  //Serial.print(sizeof(payload));
  if (result == 0) {
    Serial.println("Sent with success");
    Serial.println("...");
  }
  else {
    Serial.println("Error sending the data");
    Serial.println("...");
  }
  delay(31*1000);
}
```

5.2.3 Nodo Receptor

```
    //codigo receptor
#include <ESP8266WiFi.h>
//#include <esp_wifi.h>
//#include <WiFi.h>
#include <espnow.h>
#include <Crypto.h>
#include <SHA256.h>
//MAC Nodo Receptor: 8CxAAxB5xC5xEExD1
const char *SECRET_KEY = "n3IoTiPensoL9";

struct Data {
  float humedad;
  char Name[6];
  char hmac[32];
};

uint8_t WhiteList[][6] ={
  {0xAC, 0X0B, 0XFB, 0XD7, 0XC9, 0X02}, //Hum01
  //{0X48, 0X55, 0X19, 0X12, 0X58, 0X4C}, // //sniffer
  //{0XAC, 0X0B, 0XFB, 0XD6, 0XF0, 0XDA}, //Hum03
  {0X48, 0XE7, 0x29, 0X62, 0X70, 0X77}, // Hum02
  {0x48, 0x55, 0x19, 0x0B, 0xA2, 0xE6} //Hum05
};

//verificar hmac
bool verifyHMAC(const char *message, const char *hmacRecibido) {
  SHA256 sha256;
  sha256.reset();
  sha256.update((const uint8_t*)SECRET_KEY, strlen(SECRET_KEY));
  sha256.update((const uint8_t*)message,10);
  char hash[32];
  sha256.finalize(hash, 32);
  return (memcmp(hash, hmacRecibido, 32) == 0);
}
```

```
//funcion para validar mac
bool isMACAuthorized(uint8_t *mac) {
  for (int i = 0; i < sizeof(WhiteList) / sizeof(WhiteList[0]); i++) {
    if (memcmp(mac, WhiteList[i], 6) == 0) {
      return true; // MAC autorizada
    }
  }
  return false; // MAC no autorizada
}

void OnDataRecv(uint8_t *mac, uint8_t *incomingData, uint8_t len) {
  if(!isMACAuthorized(mac)){
    Serial.println("000");
    return;
  }
  Data data;
  memcpy(&data, incomingData, sizeof(data));
  char message[10];
  memcpy(message, data.Name, 6);
  memcpy(message +6, &data.humedad, 4);

  if(verifyHMAC(message, data.hmac)){
    Serial.printf("%s,%.2f\n",data.Name,(data.humedad));
  }
  else{
    Serial.println("000");
    return
  }
}

void setup() {
  // Init Serial Monitor
  Serial.begin(300);
  Serial.begin(115200);
  delay(500);
  while(!Serial);
```

```
  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);
  //wifi_set_macaddr(STATION_IF, &recepMAC[0]);

  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
  esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);
  esp_now_register_recv_cb(OnDataRecv);

}

void loop() {
}
```

5.2.4 Código principal unificado

```
import serial
import time
import datetime
import struct
import os
from joblib import load
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from multiprocessing import Process

# Configuración de los puertos seriales
PORT_SN = "/dev/ttyUSB0" # Puerto para el sniffer
PORT_HM = "/dev/ttyUSB1" # Puerto para el nodo de humedad
BAUD_RATE_SN = 230400
```

```
BAUD_RATE_HM = 115200

# Variable compartida para almacenar datos de humedad
dataC = pd.read_csv("pr.csv")
batch = [] # Para almacenar los grupos de paquetes que predeciremos
features = ['SA', 'Intervalo', 'RSSI', 'Canal']
modelo = load('RandomForest.joblib')
label_encoder = LabelEncoder()
columnas = ["Fecha", "DA", "SA", "Emisor", "Humedad", "Intervalo",
"FrameControl","RandomValues", "Tamano", "RSSI", "Canal"]
ruedas = dataC.copy()
print(ruedas.shape[0])

# Función para procesar datos de humedad
def leer_humedad(archivoHum):
    with open(archivoHum, "w") as filer:
        nodos = {}
        sensores = {}
        sera = serial.Serial(PORT_HM, BAUD_RATE_HM,xonxoff=False,
        rtscts=False, dsrdtr=False)
        print(f"Conectado al puerto {PORT_HM} para la recepción de humedad.")
        filer.write("Fecha,maestro,humedad\n")

        while True:
            try:
                data = sera.readline().decode().strip()
                if data == "000":
                    print("Rechazado")
                else:
                    data_ = data.split(',')
                    maestro = data_[0]
                    humedad = float(data_[1])
                    hora = datetime.datetime.now()

                    if maestro not in nodos:
                        nowTime = hora
                        nodos[maestro] = {"Past": nowTime, "Now":nowTime,
```

```
                    "Delay": nowTime - nowTime}
                else:
                    pastTime = nodos[maestro]["Now"]
                    nowTime = hora
                    nodos[maestro] = {"Past": pastTime, "Now":nowTime,
                    "Delay": nowTime - pastTime}

                delay = nodos[maestro]["Delay"].seconds
                line = f"{hora},{maestro},{humedad},{delay}\n"
                print(line)
                filer.write(line)

                if len(sensores) < 3:
                    sensores[maestro] = humedad
                else:
                    humedad_pro = sum(sensores.values()) / 3
                    sensores.clear()
                    print(f"{hora} - Humedad promedio: {round(humedad_pro, 2)}")
            except serial.SerialException as e:
                print(f"Error al leer del puerto serial: {e}")

# Función para procesar datos del sniffer
def leer_sniffer(archivoSnf):
    global ruedas, batch
    try:
        with open(archivoSnf, "w") as archivo_csv:
            nodos = {}
            ser_sn = serial.Serial(PORT_SN, BAUD_RATE_SN, xonxoff=False,
            rtscts=False,dsrdtr=False)
            print(f"Conectado al puerto {PORT_SN} para lectura de paquetes.")
            archivo_csv.write("Fecha,DA,SA,Emisor,Humedad,Intervalo,FrameControl,
            RandomValues,Tamano,RSSI,Canal\n")
            print("Cabecera escrita")

            while True:
                data = ser_sn.readline()
                data_ = ''.join(str(data)[2:-5])
```

```
            hora_local = datetime.datetime.now()

            if data:
                if b"8CAAB5C5EED1" in data:
                    data_list = data_.split(',')
                    rssi1 = data_list[0]
                    canal1 = data_list[1]
                    rate = data_list[2]
                    data_final = data_list[3]

                    frameControl = data_final[24:28]
                    frameBinario = bin(int(frameControl, 16))[2:].zfill(16)
                    bodyName = data_final[110:122]

                    if frameBinario[1:3] == "10":
                        frameControl = "Data"
                        if "48554D" in bodyName:
                            try:
                              nombre_nodo=bytes.fromhex(data_final[110:120]).
                              decode('ascii', errors='ignore')
                            except Exception as e:
                                nombre_nodo = None
                                print(f"Error al obtener el nombre del
                                nodo receptor: {e}")

                            randomValues = data_final[80:88]
                            humedadHex = data_final[102:110]

                            try:
                                byte_data = bytes.fromhex(humedadHex)
                                humedadFloat = round(struct.unpack
                                ('<f', byte_data)[0], 2)
                            except Exception as e:
                                print(f"Error al convertir {humedadHex} a float:
                                {e}")
                                humedadFloat = None
                        else:
```

```
            nombre_nodo, randomValues, humedadFloat = None, None,
            None

        elif frameBinario[1:3] == "00":
            frameControl = "Management"
            nombre_nodo, randomValues, humedadFloat = None, None, None

        elif frameBinario[1:3] == "01":
            frameControl = "Control"
            nombre_nodo, randomValues, humedadFloat = None, None, None

        else:
            print("Revisar el tipo de frame")
            frameControl = "revisar"
            nombre_nodo, randomValues, humedadFloat = None, None, None

        destinationAdd = data_final[32:44]
        sourceAdd = data_final[44:56]
        tamano = len(data_final) // 2
        rssi = convertir_Rssi(data_final[0:2])
        canal = int(data_final[2:4], 16)

        if sourceAdd not in nodos:
            nowTime = hora_local
            nodos[sourceAdd] = {"Past": nowTime, "Now": nowTime,
            "Delay": nowTime - nowTime}
        else:
            nowTime = hora_local
            pastTime = nodos[sourceAdd]["Now"]
            nodos[sourceAdd] = {"Past": pastTime, "Now": nowTime,
            "Delay": nowTime - pastTime}

        delay = nodos[sourceAdd]["Delay"].seconds
        linea = f"{hora_local},{destinationAdd},{sourceAdd},
        {nombre_nodo},{humedadFloat},{delay},{frameControl},
        {randomValues},{tamano},{rssi},
        {canal}"
```

```
                print(linea)

                try:
                    batch.append(linea)
                    archivo_csv.write(f"{linea}\n")

                    if len(batch) == 6:
                        try:
                            filas = [datos.strip().split(",") for datos in
                            batch]
                            df = pd.DataFrame(filas, columns=columnas)

                            df["Humedad"] = pd.to_numeric(df["Humedad"],
                            errors="coerce")
                            df["Intervalo"] = pd.to_numeric(df["Intervalo"],
                            errors="coerce")
                            df["RSSI"] = pd.to_numeric(df["RSSI"],
                            errors="coerce")
                            df["Canal"] = pd.to_numeric(df["Canal"],
                            errors="coerce")

                            if ruedas.shape[0] < 12:
                                df["SA"] = label_encoder.fit_transform(df["SA"])
                                df["Predict"] = modelo.predict(df[features])
                                count_zeros = (df["Predict"] == 0).sum()

                                if count_zeros > 10:
                                    print("Amenaza DoS detectada")
                                    print(df)
                                else:
                                    print("Todo normal")
                                    print(df)
                                    df = df.drop(columns=['Predict'])
                                    ruedas = pd.concat([ruedas, df],
                                    ignore_index=True)
                            else:
                                dataf = pd.concat([ruedas, df],
```

```
                                    ignore_index=True)
                                    dataf["SA"] = label_encoder.fit_transform
                                    (dataf["SA"])
                                    dataf["Predict"] =modelo.predict(dataf[features])

                                    print(dataf)
                                    count_zeros = (dataf["Predict"] == 0).sum()

                                    if count_zeros > 10:
                                        print("Amenaza DoS detectada")
                                        print(dataf)
                                    else:
                                        print("Todo normal")
                                        if ruedas.shape[0] < 132:
                                            ruedas = pd.concat([ruedas, df],
                                            ignore_index=True)

                                        dataf.to_csv("prueba.csv", index=False)
                                batch.clear()
                            except Exception as e:
                                print(f"Error procesando el batch: {e}")
                    except Exception as e:
                        print(f"Error: {e}")
    except Exception as e:
        print(f"Error procesando datos del sniffer: {e}")

# Función auxiliar para convertir valores RSSI
def convertir_Rssi(vHex):
    try:
        rssi = int(vHex, 16)
        if rssi > 127:
            rssi -= 256
        return rssi
    except Exception as e:
        print(f"Error: {e}")
        return 0
```

```
# Función principal
def main():
    hora_archivo = datetime.datetime.now().strftime("%Y-%m-%d %H-%M")
    file_humedad = f"[Hum]{hora_archivo}.csv"
    file_sniffer = f"[Sniffer]{hora_archivo}.csv"

    try:
        proceso_humedad = Process(target=leer_humedad, args=(file_humedad,))
        proceso_sniffer = Process(target=leer_sniffer, args=(file_sniffer,))

        proceso_sniffer.start()
        proceso_humedad.start()

        proceso_humedad.join()
        proceso_sniffer.join()
    except KeyboardInterrupt:
        print("\nProceso detenido por el usuario.")
    except Exception as e:
        print(f"Error: {e}")

if __name__ == "__main__":
    main()
```

BIBLIOGRAFÍA

Abunadi, I., Rehman, A., Haseeb, K., Parra, L., & Lloret, J. (2022). Traffic-aware secured cooperative framework for iot-based smart monitoring in precision agriculture. *Sensors*, *22*(17), 6676.

Adapt, C. (2023). Precision Agriculture — climate-adapt.eea.europa.eu [[Accessed 21-10-2024]].

Bhasker, B., & Murali, S. (2020). A survey on security issues in sensor cloud environment for agriculture irrigation management system. *J. Crit. Rev*, *7*(4), 1–10.

Castillo, V. S. (2023). Analysis of the scientific production on the implementation of artificial intelligence in precision agriculture. *LatIA*, *1*, 1–1.

Chaganti, R., Varadarajan, V., Gorantla, V. S., Gadekallu, T. R., & Ravi, V. (2022). Blockchain-based cloud-enabled security monitoring using internet of things in smart agriculture. *Future Internet*, *14*(9), 250.

Cloudfare. (n.d.).

Cuesta, B. A. T., & Solís, M. J. M. (2022). Agricultura de precisión y redes de sensores inalámbricos, análisis de su implementación y ventajas en el ecuador. *Serie Científica de la Universidad de las Ciencias Informáticas*, *15*(6), 54–69.

Expressif. (n.d.). ESP-NOW Wireless Communication Protocol | Espressif Systems — espressif.com [[Accessed 29-10-2024]].

Gao, Y., Doan, B. G., Zhang, Z., Ma, S., Zhang, J., Fu, A., Nepal, S., & Kim, H. (2020). Backdoor attacks and countermeasures on deep learning: A comprehensive review. *arXiv preprint arXiv:2007.10760*.

Gonzaga, R., & Sampaio, P. N. M. (2020). Mitigating man in the middle attacks within context-based sdns. *8th international workshop on ADVANCEs in ICT infrastructures and services (ADVANCE 2020)*, 1–8.

Gutiérrez Sánchez, J. P., Sánchez Sánchez, M. J., et al. (2022). Diseño e implementación de un sistema de riego para el cultivo de cacao usando el internet de las cosas (iot). *ESPOL. FIEC*.

Haseeb, K., Ud Din, I., Almogren, A., & Islam, N. (2020). An energy efficient and secure iot-based wsn framework: An application to smart agriculture. *Sensors*, *20*(7), 2081.

Hossain, M. S., Rahman, M. H., Rahman, M. S., Hosen, A. S., Seo, C., & Cho, G. H. (2021). Intellectual property theft protection in iot based precision agriculture using sdn. *Electronics*, *10*(16), 1987.

IBM. (2024). Qué es el aprendizaje por refuerzo | IBM — ibm.com [[Accessed 29-10-2024]].

Islam, M. N. U., Fahmin, A., Hossain, M. S., & Atiquzzaman, M. (2021). Denial-of-service attacks on wireless sensor network and defense techniques. *Wireless Personal Communications*, *116*, 1993–2021.

Jiang, L., Guo, Y.-C., & Yang, J.-C. (2021). Detecting anomalous quartic gauge couplings using the isolation forest machine learning algorithm. *Physical Review D*, *104*(3), 035021.

Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020). Applications of wireless sensor networks: An up-to-date survey. *Applied system innovation*, *3*(1), 14.

Karunathilake, E., Le, A. T., Heo, S., Chung, Y. S., & Mansoor, S. (2023). The path to smart farming: Innovations and opportunities in precision agriculture. *Agriculture*, *13*(8), 1593.

Keerthika, M., & Shanmugapriya, D. (2021). Wireless sensor networks: Active and passive attacks-vulnerabilities and countermeasures. *Global Transitions Proceedings*, *2*(2), 362–367.

Kour, V. P., & Arora, S. (2020). Recent developments of the internet of things in agriculture: A survey. *Ieee Access*, *8*, 129924–129957.

Lata, S., Mehfuz, S., & Urooj, S. (2021). Secure and reliable wsn for internet of things: Challenges and enabling technologies. *IEEE Access*, *9*, 161103–161128.

López, Á. H. A., Castro, J. A. Q., Perdomo, E. G., & Zambrano, A. M. M. (2022). Wsn redes de sensores inalámbricos y su aplicación a la agricultura de precisión: Un caso de metaanálisis. *Actas del Congreso de Investigación, Desarrollo e Innovación*, 245–258.

López Capó, J. (2024). Detección y mitigación de ataques de denegación de servicio en redes iot usando inteligencia artificial (ia) y técnicas de aprendizaje automático (ml).

Matsuo, Y., LeCun, Y., Sahani, M., Precup, D., Silver, D., Sugiyama, M., Uchibe, E., & Morimoto, J. (2022). Deep learning, reinforcement learning, and world models. *Neural Networks*, *152*, 267–275.

Mendoza, E., Fuentes, P., Benítez, I., Reina, D., & Núñez, J. (2020). Red de sensores inalámbricos multisalto para sistemas domóticos de bajo costo y área extendida. *Revista Iberoamericana de Automática e Informática industrial*, *17*(4), 412–423.

Microsoft. (n.d.). HMACSHA256 Clase (System.Security.Cryptography) — learn.microsoft.com [[Accessed 12-11-2024]].

Mouha, R. A. R. A., et al. (2021). Internet of things (iot). *Journal of Data Analysis and Information Processing*, *9*(02), 77.

Muñoz, J. M., & Castañeda, R. (2023). The use of machine learning in volatility: A review using k-means. *Revista Universidad y Empresa*, *25*(44).

NCSC. (2016). Denial of Service (DoS) guidance — ncsc.gov.uk [[Accessed 21-10-2024]].

Palacios, L. E. S., Alcivar, F. R. M., Sánchez, S. T. T., Montes, A. C. L., & Guajala, G. N. T. (2024). Agricultura de precisión en el ecuador. *Ciencia Latina Revista Científica Multidisciplinar*, *8*(1), 1532–1542.

Pasic, R., Kuzmanov, I., & Atanasovski, K. (2021). Esp-now communication protocol with esp32. *Journal of Universal Excellence*, *6*(1), 53–60.

Prodanović, R., Rančić, D., Vulić, I., Zorić, N., Bogićević, D., Ostojić, G., Sarang, S., & Stankovski, S. (2020). Wireless sensor network in agriculture: Model of cyber security. *Sensors*, *20*(23), 6747.

Puente Fernández, J. A. (2021). Técnicas de monitorización en transmisiones multimedia para redes definidas por software.

Romero Amondaray, L., Artigas Fuentes, F. J., & Calderón, C. A. (2020). Redes de sensores inalámbricos definidas por software: Revisión del estado del arte. *Ingeniería Electrónica, Automática y Comunicaciones*, *41*(2), 39–50.

Sharma, H., Shajahan, B., Elangovan, R., & Thirumalaisamy, M. (2022). Dos attack detection mechanism in wireless sensor networks. *Salud, Ciencia y Tecnología*, (2), 244.

Sishodia, R. P., Ray, R. L., & Singh, S. K. (2020). Applications of remote sensing in precision agriculture: A review. *Remote sensing*, *12*(19), 3136.

Tobar, M. (2022). Agricultura de precisión y redes de sensores inalámbricos, análisis de su implementación y ventajas en el Ecuador — dialnet.unirioja.es [[Accessed 27-10-2024]].

TORRES VENTURA, J., RUELAS PUENTE, A., & HERRERA GARCIA, J. (n.d.). Rendimiento para la interoperabilidad entre raspberry pi, esp8266 y plc con node-red para el iiot. ingenius [online]. 2023, n. 29.

Triantafyllou, A., Sarigiannidis, P., & Bibi, S. (2019). Precision agriculture: A remote sensing monitoring system architecture. *Information*, *10*(11), 348.

Tuli, R. (2020). Packet Sniffing and Sniffing Detection [[Accessed 25-11-2024]].

Unir. (2023). ¿Qué es un sniffer de red? [[Accessed 11-11-2024]].