UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERIA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

APLICACIÓN DE CRIPTOGRAFÍA DE CURVA ELÍPTICA PARA REDES DE SENSORES EN SISTEMAS DE ALERTA TEMPRANA

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TITULO DE INGENIERO EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

AUTOR: ANDREA ISABEL ORTEGA PATIÑO

DIRECTOR: MSC. FABIÁN GEOVANNY CUZME RODRÍGUEZ

Ibarra-Ecuador

2023

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	0401688916		
APELLIDOS Y NOMBRES:	Ortega Patiño Andrea Isabel		
DIRECCIÓN:	Ibarra, Calle Esmeraldas 2-317 y 13 de Abril		
EMAIL:	aiortega@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0995403247

DATOS DE LA OBRA	
TÍTULO:	APLICACIÓN DE CRIPTOGRAFÍA DE CURVA ELÍPTICA PARA REDES DE SENSORES EN SISTEMAS DE ALERTA TEMPRANA
AUTOR (ES):	Ortega Patiño Andrea Isabel
FECHA: DD/MM/AAAA	20/07/2023
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	☒ **PREGRADO** ☐ **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniería en Electrónica y Redes de Comunicación
ASESOR /DIRECTOR:	MSC. Fabián Cuzme Rodríguez

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 20 días del mes de julio del 2023

EL AUTOR:

Andrea Ortega

.....

Andrea Isabel Ortega Patiño

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCAS APLICADAS

CERTIFICACIÓN:

MAGISTER FABIÁN CUZME RODRÍGUEZ, DIRECTOR DEL PRESENTE TRABAJO DE TITULACIÓN CERTIFICA:

Que, el presente trabajo de titulación "APLICACIÓN DE CRIPTOGRAFÍA DE CURVA ELÍPTICA PARA REDES DE SENSORES EN SISTEMAS DE ALERTA TEMPRANA" ha sido desarrollado por la señorita Andrea Isabel Ortega Patiño bajo mi supervisión.

Es todo en cuanto puedo certificar en honor de la verdad.

Ing. Fabián Geovanny Cuzme Rodríguez, MsC.

DIRECTOR

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

DEDICATORIA

Quiero dedicar este trabajo de titulación a mis padres que siempre están brindándome su apoyo, porque toda mi vida han estado junto a mí, en cada proyecto, en cada sueño, en cada dificultad, siempre me han demostrado su amor incondicional, y me han impulsado a salir adelante. Ellos han anhelado la culminación de este proyecto tanto como yo, se lo merecen.

A mi esposo Cristian, que desde el primer momento supo acompañarme, comprenderme y animarme para continuar en este camino hasta finalizar.

A mi hija, Lía, por enseñarme que el amor es lo que mueve todo lo que conocemos y es por lo que existimos, y sobre todo por ser el motor que me impulsa a salir adelante.

Andrea Isabel Ortega

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

AGRADECIMIENTO

Agradezco a Dios por bendecirme con este proyecto y por todo lo que ha puesto en mi vida, gracias a Él y su gracia divina he podido lograr mis propósitos siempre caminando de su mano.

Agradezco también a mis padres que han puesto su esfuerzo, su empeño, y dedicación para que yo pueda lograr mis objetivos, y, sobre todo, por confiar en mis capacidades siempre. Gracias a mis hermanos por su apoyo incondicional, por sus palabras que siempre fueron de ánimo y de fortaleza para continuar en la búsqueda de mis objetivos.

Agradezco a mi esposo Cristian, por estar siempre apoyándome y animándome, por su entrega, su confianza, y su ayuda incondicional para que yo pueda lograr esta meta.

Agradezco a mi director MSC. Fabián Cuzme, por su colaboración, su tiempo, sus aportes a esta investigación y sobre todo por su paciencia y apoyo para lograr culminar este proyecto. Al MSC. Jaime Michilena, asesor de este proyecto, le agradezco su comprensión, su apoyo, su colaboración, y a todos los docentes quienes durante toda la carrera universitaria marcaron mi formación compartiendo no solamente conocimientos y experiencias sino también valores éticos que siempre tendré presentes en mi vida profesional y personal.

A mis amigos y compañeros que han estado siempre pendientes brindándome su apoyo y sus palabras de aliento.

Andrea Isabel Ortega

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

RESUMEN

Las Redes de Sensores Inalámbricas (WSN) ya forman parte de la vida cotidiana del ser humano, convirtiéndose así en una herramienta bastante útil en diversas actividades, entre las más utilizadas se encuentran los Sistemas de Alerta Temprana (SAT), los cuales usan las redes de sensores con la finalidad de optimizar la forma en que las personas podrían prevenir desastres o al menos disminuir su impacto en el medio ambiente o en el ambiente urbano – residencial.

Las WSN, como todas las redes que generan, trasmiten y procesan datos, están expuestas a diversas amenazas en cuanto a seguridad, pues si bien en este tipo de redes las capacidades de software y hardware es mínima, también es importante proteger la integridad y privacidad de lo que se trasmite. Uno de los métodos de encriptación a tomar en cuenta es la criptografía de curvas elípticas (ECC), ya que para realizar el cálculo de generación de claves y cifrado no requiere tantos recursos en comparación con otros esquemas como RSA. En esta investigación se presenta la aplicación de ECC en una red WSN con la finalidad de mostrar los efectos que genera la aplicación de este tipo de seguridad en la red haciendo uso de simuladores de red como OMNet++ y MatLab.

Es importante para una WSN que los procesos adicionales como en este caso la seguridad de los datos, no generen mayor costo en cuanto a recursos, por esta razón se demuestra que al incorporar un algoritmo criptográfico de curvas elípticas el delay de la red no se ve afectado, además la carga computacional no indica altos valores de consumo para los dispositivos.

En este campo de investigación es importante tomar en cuenta el constante desarrollo de nuevas técnicas de criptografía, lo que da lugar a la necesidad de realizar pruebas para encontrar el método de seguridad más eficiente para una WSN, dentro de la ECC actualmente se desarrollan algunas variantes del esquema que pueden adaptarse a cualquier tipo de red incluyendo principalmente las WSN.

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

ABSTRACT

Wireless Sensor Networks (WSN) are already part of the daily life of the human being, thus becoming a very useful tool in various activities, among the most used are Early Warning Systems (SAT), which use sensor networks in order to optimize the way in which people could prevent disasters or at least reduce their impact on the environment or in the urban-residential environment.

WSNs, like all networks that generate, transmit and process data, are exposed to various security threats, because although software and hardware capabilities are minimal in this type of network, it is also important to protect the integrity and privacy of what is transmitted. One of the encryption methods to consider is elliptic curve cryptography (ECC), since it does not require as many resources to perform the calculation of key generation and encryption compared to other schemes such as RSA. In this research, the application of ECC in a WSN network is presented in order to show the effects generated by the application of this type of security in the network using network simulators such as OMNet++ and MatLab.

It is important for a WSN that additional processes, such as data security in this case, do not generate a higher cost in terms of resources, for this reason it is shown that by incorporating an elliptic curve cryptographic algorithm, the network delay is not affected, and the computational load does not indicate high consumption values for the devices.

In this field of research, it is important to consider the constant development of new cryptography techniques, which gives rise to the need to carry out tests to find the most efficient security method for a WSN. Within the ECC, some variants of the scheme are currently being developed that can be adapted to any type of network, including mainly WSNs.

CONTENIDO

Índice de Figuras

Índice de Tablas

Índice de Ecuaciones

CAPÍTULO I

Antecedentes

En este capítulo se procura describir los antecedentes en base a los cuales se ha originado la problemática del presente trabajo de investigación. Adicionalmente se presenta la justificación por la cual se ha realizado el proyecto juntamente con el alcance de este.

1.1 Problema

Las redes de sensores han tenido una gran expansión, llegando a formar parte de las actividades cotidianas del ser humano en diversos ámbitos en los que se pueden destacar: Información de respuesta ante emergencias, con sensores que recopilan información para ser enviada a las entidades de respuestas a emergencias; otras aplicaciones implican el monitoreo de la salud, gestión de la energía, alerta temprana, etc.

En las redes de sensores, la información requiere una combinación de autenticación, integridad, privacidad y seguridad, en especial cuando se trata de redes de sensores en sistemas de alerta temprana, en los que es necesario utilizar métodos y herramientas que garanticen los requerimientos de seguridad y además de disponibilidad del sistema, pues no resulta factible implementar seguridad en un sistema afectando al rendimiento de este. En las redes de sensores de los sistemas de alerta temprana la baja latencia se convierte en un requerimiento esencial por lo que es necesario usar mecanismos de seguridad con los cuales los retardos no se vean afectados. Pero realmente no se puede cubrir estas necesidades en la información; a consecuencia de que, en este tipo de redes, los sensores tienen una capacidad limitada en cuanto a procesamiento, por lo que no son adecuados para realizar implementaciones de seguridad muy robustas.

A nivel local existen redes de sensores en las que no se h implementado ningún tipo de seguridad, tal es el caso de la red de sensores desplegada en la loma de Guayabillas en la ciudad

de Ibarra, en la que se ve necesaria la implementación de un esquema de seguridad por tratarse de un sistema de alerta temprana, pero que a su vez no requiera de grandes capacidades de procesamiento en los nodos.

Si bien existen métodos de encriptación con alto rendimiento y altas capacidades para sistemas computacionales, como son los algoritmos de cifrado RSA, Diffie-Hellman, funciones hash, etc., en las redes de sensores existen otro tipo de requerimientos; lo que se intenta es llegar a una solución de seguridad que no consuma muchos recursos de la red. Por esta razón se desea investigar sobre un nuevo algoritmo basado en la curva elíptica para las redes de sensores, de manera que, al implementar este algoritmo de seguridad en la simulación de una red de sensores, se pueda comprobar que cumple los requerimientos de seguridad sin afectar al funcionamiento de la red ni a consumir muchos recursos de ésta.

1.2 Objetivos

1.2.1 Objetivo General

Implementar un algoritmo de encriptación de curva elíptica para mejorar la seguridad de la información en redes de sensores en sistemas de alerta temprana.

1.2.2 Objetivos Específicos

- Analizar la bibliografía necesaria para determinar las características y funcionamiento de la encriptación de curva elíptica.
- Determinar los requerimientos funcionales para implementar un algoritmo de criptografía de curva elíptica.
- Definir un escenario real de implementación de la red de sensores para tomarlo como base en las pruebas de funcionamiento del algoritmo.

- Evaluar el funcionamiento del algoritmo de curva elíptica utilizando una red de sensores simulada en un software de simulación.

1.3 Alcance

Para este proyecto se realizará un análisis de la bibliografía seleccionada para tener claro el funcionamiento y requerimientos de los sensores y principalmente de la encriptación de curva elíptica, con los respectivos parámetros que intervienen para calcular el algoritmo.

Con base en lo analizado se determinará los requerimientos necesarios para poder implementar el algoritmo de curva elíptica; tomando como base metodológica la norma IEEE 29148, que proporciona un tratamiento de procesos de ingeniería aplicados a todo tipo de software.

Tomando como base una red de sensores de alerta temprana para incendios implementada en la Loma de Guayabillas, se realizará la simulación de una red en condiciones similares utilizando el software de simulación. Uno de los parámetros establecidos por la red existente es un ancho de banda mínimo requerido para la transmisión de 2 448 Kbytes, este es un parámetro importante por considerar para poder determinar en qué medida el algoritmo afecta a la latencia de la red. Además, los protocolos a usarse serán determinados por la tecnología ZigBee.

Para comprobar los efectos de la implementación del algoritmo en la red de sensores de alerta temprana, se verificará que la latencia de la red se mantenga, comprobando el tiempo que le toma al sensor recabar y enviar la información, como el tiempo que tarda la información en llegar al destino, para posteriormente implementar el algoritmo y realizar nuevamente la verificación de latencia; esto permitirá evidenciar el retardo que introduce la implementación de la encriptación de curva elíptica. De esta manera se podrá demostrar que cumple los principales fines de la investigación que son comprobar que este tipo de criptografía no afecta

sobremanera en la latencia de la red y, por otra parte, provee un mecanismo de seguridad de la información más robusto.

1.4 Justificación

Dentro de los ámbitos donde se han desplegado las redes de sensores vemos que en la mayoría de los casos no se han manejado implementaciones de seguridad, tal es el caso del trabajo **"Diseño de una red inalámbrica para una WSN de un sistema de alerta temprana de incendios para el Bosque Protector Guayabillas"** realizado por Vanessa Enríquez (Enríquez & Michilena, 2018) por mencionar un ejemplo en el que en la red de sensores no se ha implementado ningún tipo de seguridad, teniendo en cuenta que la información que se transmite en este caso es de vital importancia para contribuir a la conservación del medio ambiente. En este contexto se propone la investigación de un algoritmo de curva elíptica para implementar seguridad en la información transmitida, con la finalidad de mantener los estándares de seguridad, latencia y rendimiento de la red de sensores.

En base al enlace que se tomará como referencia para realizar las pruebas del algoritmo de seguridad, se puede observar que, al ser un enlace inalámbrico, requiere de un esquema robusto de seguridad, para que la transmisión de la información no sea alterada por terceros

Según un estudio realizado por Dhillon Ksaur Parwinder denominado *"Elliptic Curve Cryptography for Real Time Embedded Systems in IoT Networks"* el algoritmo de curva elíptica puede ayudar a colocar una seguridad más robusta en las redes de sensores; además, es de 5 a 15 veces más rápida en comparación con la clave privada RSA (Parwinder, 2015).

Por consiguiente, se quiere indagar en esta temática para ser implementada a nivel local, en un software que simule redes de sensores para ver su funcionalidad y demostrar que el algoritmo de curva elíptica no requiere de mucho procesamiento para la encriptación, y por ende lo hace el más apto para implementarse en redes de sensores.

Para la simulación de la red de sensores se prevé hacer uso de un software simulador de redes, usado en el ámbito de la investigación y principalmente en el área educativa. Este simulador es un software libre que permite la interconexión de dispositivos tanto para redes cableadas como inalámbricas, y utiliza C++ como lenguaje de programación. Gracias a que permite desarrollar modelos de simulación de alto desempeño es considerado como un emulador, soportando algunos tipos de protocolos de comunicación como son: IP, LTE, Wi-Fi, etc., lo que permitirá la simulación de un enlace para la comprobación pertinente a esta investigación.

CAPÍTULO II

Fundamentación Teórica

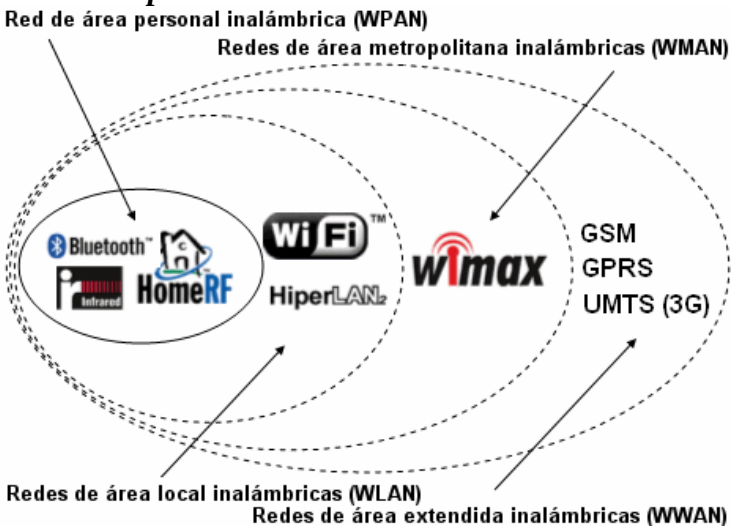
En este capítulo se realiza un análisis de la bibliografía relacionada con las redes inalámbricas, las redes de sensores inalámbricas, además de la seguridad en estas redes y de forma enfática la criptografía de curva elíptica ECC.

2.1 Redes Inalámbricas

Las redes inalámbricas son redes que permiten transmitir información entre dispositivos por medio de ondas de radio, de esta manera no se requiere cableado para la comunicación. Tanto para la transmisión como para la recepción de datos utilizan antenas.

Como ventaja frente a otros sistemas, las redes inalámbricas son más sencillas de instalar, debido a que no se requiere cablear, además brindan movilidad, por lo que los dispositivos pueden trasladarse de un lugar a otro, sin presentar inconvenientes en la comunicación, siempre que se encuentren dentro del radio de alcance del emisor. En la Figura 1 se observa la clasificación de las redes inalámbricas según su alcance.

Figura 1
Tipos de Redes inalámbricas por su alcance.



Tomado de "Modelo de cobertura para redes inalámbricas de interiores" por Camargo Olivares, J., & Hornillo Mellado, S. (2009).

2.1.1 Redes Inalámbricas Según Su Alcance

2.1.1.1 WPAN (Wireless Personal Area Network). Es una tecnología inalámbrica que proporciona conectividad entre dispositivos de radio portátiles de corto alcance, no más de 10 metros. Por lo general trabajan a una frecuencia de 2,4 GHz. Los dispositivos de este tipo de redes son los típicos del usuario como un teléfono celular, dispositivos de audio o cámaras fotográficas. La tecnología más usada es Bluetooth (IEEE 802.15.1)

2.1.1.2 WLAN (Wireless Local Area Network). Es una solución rentable en lo que respecta al acceso inalámbrico a Internet, capaz de satisfacer la mayoría de los requisitos de comunicación actuales. (Bellalta, 2016) La tecnología más conocida es Wi-Fi (IEEE 802.11ax), y por lo general se complementa con una LAN cableada para brindar al usuario un considerable ancho de banda y a la vez movilidad.

2.1.1.3 WMAN (Wireless Metropolitan Area Network). Este tipo de red es una forma de red inalámbrica que tiene un área de cobertura de aproximadamente el tamaño de una ciudad. Se trata de redes, generalmente, punto a punto o punto a multipunto con enlaces individuales. WiMAX es la forma de WMAN más utilizada.

2.1.1.4 WWAN (Wireless Wide Area Network). Este tipo de redes inalámbricas pueden permitir la comunicación con áreas de coberturas mayores a las de una ciudad, por ejemplo, se puede mencionar las comunicaciones móviles en todas sus tecnologías, en las que se puede establecer una comunicación a varios kilómetros de distancia.

Además, en las redes inalámbricas, existen diversas tecnologías que permiten el envío y recepción de información de un punto a otro, a continuación, se menciona algunas de ellas:

2.1.2 Tecnologías En Redes Inalámbricas

Las redes inalámbricas por lo general utilizan radiofrecuencia para la comunicación entre dispositivos, incluyendo una banda de frecuencias específica que depende del área de aplicación, estos parámetros están definidos en las diferentes tecnologías de comunicación existentes para este tipo de redes, a continuación, se describe el funcionamiento y características de las tecnologías inalámbricas más utilizadas.

2.1.2.1 IEEE 802.11x (Wireless LAN, Wi-Fi). Wi-Fi es una tecnología de red inalámbrica que permite que dispositivos como computadoras (computadoras portátiles y de escritorio), dispositivos móviles (teléfonos inteligentes y dispositivos portátiles) y otros equipos (impresoras y cámaras de video) interactúen con Internet. Permite que estos dispositivos, y muchos más, intercambien información entre sí, creando una red. (CISCO, 2017)

La conexión de los dispositivos se realiza utilizando un router inalámbrico en el cual se configura las condiciones de trabajo de la tecnología y la seguridad de la red. Esta tecnología incluye los estándares IEEE 802.11 a/b/g y por lo general alcanza un área de cobertura de 100 metros con una tasa máxima de señal de 54 Mb/s. (Lee et al., 2017)

2.1.2.2 Bluetooth (IEEE 802.15.1). Es una tecnología basada en un sistema inalámbrico de radio que utiliza dispositivos de bajo costo en el mercado que se caracterizan por su corto alcance perteneciendo así a la gama de tecnologías de las redes tipo WPAN. Bluetooth funciona con el mecanismo maestro – esclavo, mientras los esclavos pueden comunicarse con su maestro únicamente con una conexión punto a punto según lo determine el maestro, éste a su vez puede iniciar una comunicación punto a punto o bien, punto a multipunto en dirección a los esclavos.

Como ya se mencionó, Bluetooth es de corto alcance no superando los 10 metros de distancia en la comunicación, y la tasa de señal es de máximo 1 Mb/s. (Lee et al., 2017)

2.1.2.3 Zigbee (IEEE 802.15.4). Al igual que Bluetooth define características de transmisión de bajas velocidades, pero con un consumo mínimo de energía y un rango de cobertura de 10 – 100 metros (Lee et al., 2017). Los dispositivos usados en esta tecnología se han definido de dos tipos: dispositivos físicos y dispositivos lógicos.

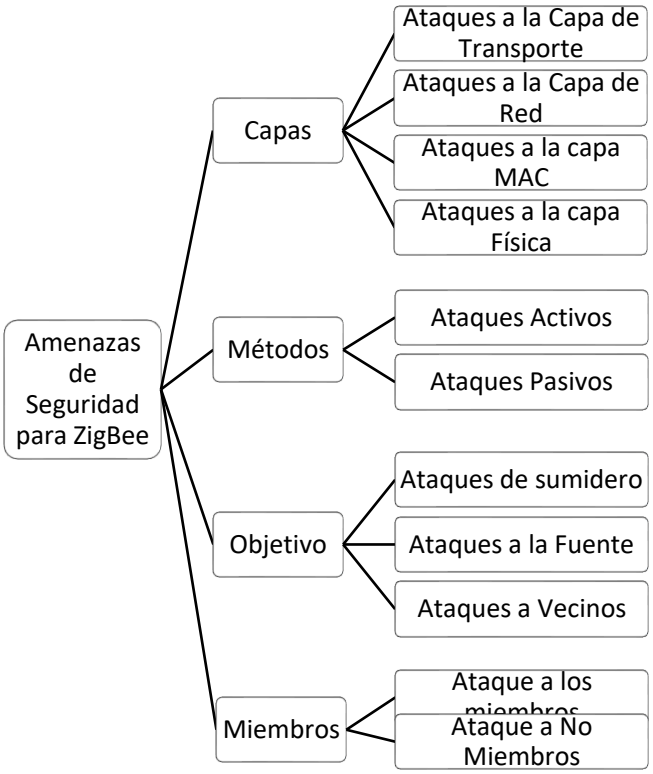
Los dispositivos de tipo físico, a su vez, se clasifican en dos tipos: dispositivo de función completa (FFD) y dispositivo de función reducida (RFD). Cualquier dispositivo puede actuar como nodo sensor, nodo de control y dispositivo compuesto independientemente de su tipo. Dependiendo de su ubicación en la red, los FFD pueden tener uno o más dispositivos secundarios y realizan funciones de enrutamiento para estos dispositivos secundarios. Los RFD no realizan la función de enrutamiento en una red y, por lo tanto, no pueden tener ningún dispositivo secundario.

Los dispositivos de tipo lógico se clasifican además en tres tipos: coordinador, enrutador y dispositivo final. Entre estos dispositivos lógicos, el coordinador es el dispositivo más capaz, que forma la raíz del árbol de la red. Debe haber exactamente un coordinador de ZigBee en una red para iniciar la formación de un árbol de red. También actúa como puente

hacia otras redes. Los dispositivos finales ZigBee poseen una funcionalidad limitada para comunicarse solo con un coordinador o un enrutador; no puede transmitir datos para otros dispositivos. Debido a esta funcionalidad limitada, los dispositivos finales pueden "dormir" durante una cantidad significativa de tiempo y, por lo tanto, pueden disfrutar de una larga vida útil. (Obaid et al., 2016)

Entre las principales ventajas de esta tecnología se puede notar la interoperabilidad del producto, además se puede esperar mayor innovación en la tecnología si en la actualidad ya permite la conectividad entre más de 100 dispositivos, además como ya se ha mencionado esta tecnología puede garantizar la duración de las baterías de los dispositivos debido a su bajo consumo de energía, esto es ideal para la implementación de redes de sensores en las cuales no siempre se cuenta con conexiones a fuentes de energía para todos los dispositivos.

Figura 2
Ataques y amenazas en el estándar ZigBee.



Tomado de “ZigBee Security Vulnerabilities: Exploration and Evaluating” por (Khanji et al., 2019)

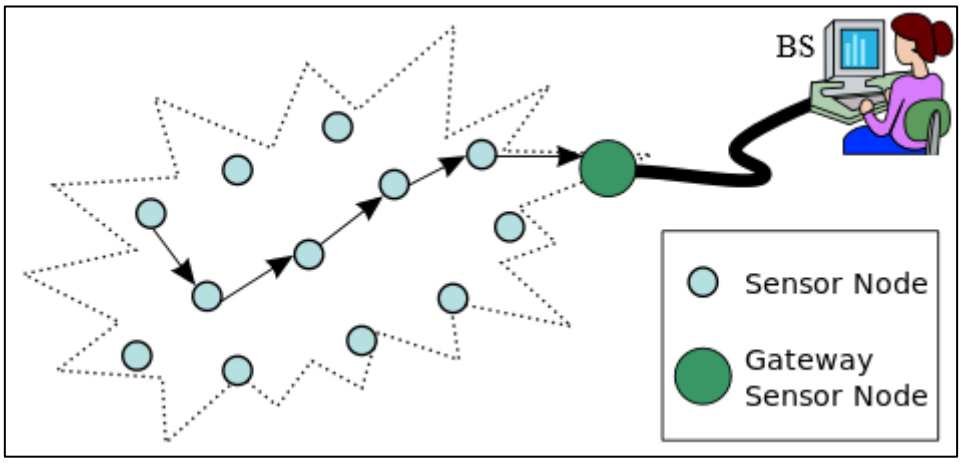
2.1.3 Redes De Sensores Inalámbricas (WSN)

Una red inalámbrica de sensores o WSN se compone de diversos dispositivos pequeños, autónomos, que se distribuyen estratégicamente según la finalidad denominados nodos de sensores. Estos se instalan por lo general, en torno a un fenómeno con el fin de realizar un monitoreo de diversos eventos a suceder y que a su vez tienen la capacidad de almacenar y transmitir la información de la red utilizando tecnologías inalámbricas.

La aplicación de este tipo de redes es muy amplia en distintos campos, entre ellos, agricultura, medicina, domótica, medio ambiente, de manera que las personas interactúan cada vez más con este tipo de sistemas que facilitan las actividades cotidianas de los seres humanos.

Como una ventaja de este tipo de redes, los protocolos de comunicación permiten que el nodo consuma lo mínimo de energía, ya que así en modo pasivo, es decir, cuando no envían datos, los sensores no consumen energía.

Figura 3
Elementos de una WSN.



Tomado de "Prototipo de Parqueadero Inteligente mediante Red de Sensores" por Márquez M., Lara R., Gordillo R., 2014

Además, son unidades autónomas, que por sí solos intercambian el estado de consumo de energía según su actividad, y pueden alimentarse con baterías pequeñas.

2.1.3.1 Elementos De Una WSN. Una WSN está conformada por tres elementos principales como se muestra en la Figura 3 y se describen a continuación:

- Nodos sensores
- Gateway
- Estación Base

Nodos Sensores.

También denominados “motas” debido a su tamaño reducido, se trata de dispositivos encargados de captar información del medio para procesarla y luego enviarla hacia otro punto de la red. Los nodos se configuran para ser parte de la red con un objetivo, es decir, no puede existir un nodo aislado, siempre deben ser parte del sistema. La arquitectura de un nodo sensor es muy poco compleja y consta de alimentación, Comunicación inalámbrica, Procesador, Sensores y Memoria.

Gateway

También es un nodo, pero tiene características especiales, pues se encarga de unir dos redes de diferente tipo, es decir la conexión entre la red de sensores y una red de datos, para lo cual debe configurarse de manera que tenga conectividad con todos los elementos de la red de sensores y así pueda enviar la información recolectada por la red de datos.

Estación Base.

Por lo general es un ordenador, conectado al Gateway en el cual se recibe toda la información de la red de sensores y se realiza el tratamiento de esta, desde aquí se realiza el monitoreo del entorno estudiado y el almacenamiento de los datos para su estudio y determinación de decisiones respecto a este.

2.1.3.2 Limitaciones de las redes de sensores. Las limitaciones que tienen los nodos en lo que respecta a capacidad computacional, almacenamiento, ancho de banda y energía, llegan a impedir que las WSN cumplan con la demanda que exigen las tecnologías IoT, además es necesario mantener los nodos funcionales todo el tiempo y lograr mayor eficiencia de los sistemas. Por esta razón se requiere encontrar la mejor solución a todas estas problemáticas de manera que la red funcione adecuadamente por un tiempo prolongado.

Energía.

Los nodos sensores generalmente funcionan con una vida útil limitada, y a pesar de que la solución a esto ha sido en la mayoría de los casos usar baterías recargables, este problema sigue siendo prioridad en el despliegue de redes de sensores. EL problema radica esencialmente en la vida útil de las baterías y por ende en la vida útil de la WSN. En (Banđur et al., 2019) se define la vida útil de un nodo sensor como el tiempo de funcionamiento del nodo sin necesidad de ninguna intervención externa, como el reemplazo de la batería.

Otro problema está relacionado con el impacto que genera la vida útil de la batería en el rendimiento del nodo, pues si la vida útil de la batería empieza a finalizar, esto disminuye el rendimiento del nodo sin necesidad de dejar de funcionar, es decir, en algunos casos admiten solo unas funcionalidades y otras no, por ejemplo, podrán enviar solicitudes, pero no pueden recibir; o también, entregar información de manera errónea.

Por esta razón la gestión eficiente de la energía implica una garantía para prolongar la vida útil y el adecuado funcionamiento de la red. El consumo de energía varía de acuerdo con el campo de aplicación del sensor, según la cantidad de información que es capaz de transmitir por unidad de tiempo (Banđur et al., 2019), por lo que es necesario tomar esto en cuenta al momento de diseñar una red de sensores.

Capacidad.

En cuanto al procesamiento, se observa la necesidad de una adecuada gestión de la red de sensores, tomando en consideración la capacidad de cada uno de los elementos del nodo, de manera que le permita a éste brindar la información con un tiempo de respuesta aceptable para este tipo de redes. (Dender-Zurita et al., 2018)

En lo que respecta al monitoreo ambiental, algunas limitaciones son la cobertura, la atenuación de la señal, la configuración y las dificultades operativas. Además, las áreas con terrenos montañosos dificultan la transmisión de radio. Al mismo tiempo es necesario un mantenimiento permanente para aumentar el tiempo de vida útil del sistema en general.(Báez, 2017)

2.2 Seguridad De Redes

Es importante mencionar que en una red de comunicación de cualquier tipo se requiere cierto nivel de seguridad, especialmente de la información, pues conociendo la vulnerabilidad de los datos que se transmiten, es necesario aplicar técnicas y métodos para proteger esa información.

2.2.1 Seguridad De La Información

Es un término que encierra diversas medidas de seguridad que se aplican a la información de cualquier tipo, independientemente de cómo se almacene o se transmita. Por tanto, se trata de establecer un conjunto de procedimientos cuyo objetivo principal es proteger la integridad y la confidencialidad de la información

La Seguridad informática, por su parte, especifica las medidas y procedimientos necesarios para proteger la información almacenada en medios informáticos o digitales. Este tipo de seguridad consta de tres principios básicos con el objeto de proteger la información, estos son Confidencialidad, Integridad y Disponibilidad.

Confidencialidad implica el hecho de garantizar que la información transmitida no sea escuchada/leída por intrusos, esto se logra aplicando algunos métodos de seguridad entre los que destaca el cifrado, el mensaje es modificado utilizando algoritmos de tal manera que solamente el receptor pueda descifrarlo y así acceder a la información. (Salazar, 2016)

En cuanto a integridad se refiere, el objetivo principal es que la información llegue completa al destinatario, sin ninguna alteración en el trayecto. Para lograrlo existen métodos de encriptación encargados de realizar la comprobación de integridad del mensaje, los más usados son los denominados funciones hash.(Salazar, 2016)

La disponibilidad se refiere a la capacidad del sistema para mantener el acceso a la información siempre que se requiera, sin pérdidas de datos y además sin bloqueos tanto de la red como del acceso de los usuarios permitidos. (López Grande & Guadrón Gutiérrez, 2015)

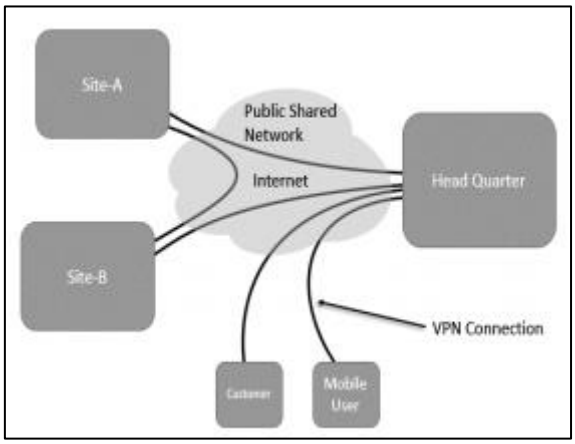
2.2.2 Seguridad En Redes Inalámbricas

En general las redes de datos están propensas a sufrir ataques de intrusos para escuchar, leer e incluso alterar la información que se transmite, pero en las redes inalámbricas este riesgo es aún más alto, pues las redes cableadas pueden protegerse en el emisor y en el receptor, en cambio las redes inalámbricas por su medio de transmisión que son ondas electromagnéticas en el aire, presentan mayor dificultad para ser protegidas de intrusiones en la transmisión de la información de un extremo al otro.

En seguridad de redes inalámbricas existen algunos estándares y tecnologías que, según su campo de acción y aplicación, permiten garantizar que los datos transmitidos no puedan ser leídos ni alterados por terceros, algunos de ellos son:

2.2.2.1 VPN. Es una arquitectura de red que se implementa a través de una red pública para respaldar la privacidad en una red pública compartida **(Singh & Gupta, 2016a)**, esta es una solución viable y muy rentable para las organizaciones, ya que ahorra el gran costo de infraestructura de insertar un modelo de seguridad robusto frente a la navegación por Internet. En la Figura 4 se muestra el escenario típico de una VPN, en donde el protocolo de tunelización garantiza un transporte seguro para todos los servicios de red.

Figura 4
Escenario Típico de VPN



Tomado de "A new approach for the security of VPN" por (Singh & Gupta, 2016b)

VPN utiliza protocolos de tunelización para respaldar su funcionamiento. Estos protocolos proporcionan un modo de transporte de datos seguro para todos los servicios de red. De esta manera se establece un canal lógico seguro para una comunicación virtual punto a punto o punto multipunto, que encapsula el datagrama IP para ocultar la información original de posibles intrusos. Generalmente una VPN utiliza el estándar de cifrado de datos DES, AES y el algoritmo Blowfish (Singh & Gupta, 2016b).

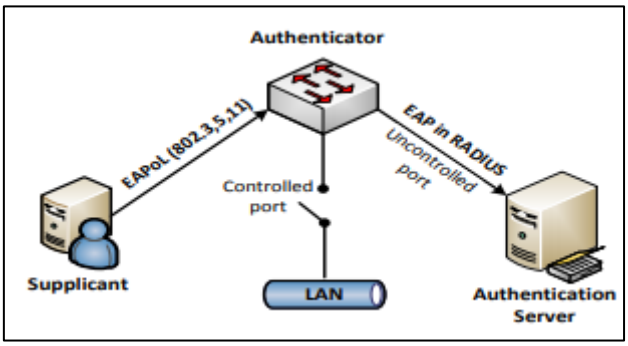
2.2.2.2 802.1X. El objetivo del estándar IEEE 802.1x es restringir los hosts no autorizados a acceder a la red, esto lo hace mediante tres aspectos que son los siguientes:

Solicitud. Es el dispositivo que actúa como cliente e intenta obtener acceso a la red inalámbrica por medio del uso de credenciales.

Autenticación. Suele ser un conmutador o punto de acceso inalámbrico, ubicado entre el cliente y el servidor de autenticación. En este componente de la red se reenvía la información de autenticación del cliente al Servidor.

Servidor de autenticación. Por lo general, se trata de un servidor RADIUS que autentica uno o varios clientes según credenciales proporcionadas, las cuales al llegar al servidor se comparan con una base de datos para poder verificar la identidad, un ejemplo de estas bases de datos es un servidor LDAP (Hu et al., 2016).

Figura 5
Componentes de la trama 802.1x.



Tomado de "Performance analysis of microsoft network policy server and free RADIUS authentication systems in 802.1x based secured wired ethernet using PEAP" por (Chughtai et al., 2019).

2.2.2.3 Portal Cautivo. Para mejorar la seguridad en la red, existe un mecanismo más usado que la autenticación WPA2 en el caso de Wi-Fi, se trata de los portales cautivos.

Un portal cautivo utiliza una página web para autenticar a los usuarios, si un cliente desea acceder a Internet, el navegador web le redirige a una página de inicio donde debe introducir las credenciales respectivas de inicio de sesión (Aryeh et al., 2016). Una ventaja muy representativa del uso de portales cautivos es que el cliente no requiere la instalación de ningún tipo de software en su dispositivo de acceso, tampoco debe realizar configuraciones adicionales, lo que hace que el acceso a la red sea menos complicado.

2.2.2.4 Protocolos De Seguridad En Redes Inalámbricas. Para que los estándares y tecnologías de seguridad de redes puedan funcionar adecuadamente, existen algunos protocolos que juegan un papel de soporte a las características de seguridad en redes inalámbricas, la autenticación es un punto clave para el desarrollo de algunos de los protocolos que se mencionan a continuación.

PPP

Es un protocolo que proporciona un método estándar para encapsular la información del protocolo de capa de red a través de enlaces punto a punto. PPP también define un protocolo de control de enlace extensible y propone una familia de protocolos de control de red (NCP) para establecer y configurar diferentes protocolos de capa de red. (IETF, 1992)

PPTP

Es un protocolo de red que permite la transferencia segura de datos desde un cliente remoto a un servidor privado configurando una VPN sobre TCP/IP. La tecnología de PPTP es una extensión del protocolo punto a punto de acceso remoto (PPP) emitido por el IETE (Mufida et al., 2017). PPTP convierte los paquetes PPP en datagramas IP para que puedan ser enviados. La principal ventaja de este protocolo es que puede ser usado una PSTN para construir una VPN incluso sobre esta plataforma.

IPSec

Es un protocolo de túnel VPN de los más seguros y confiables puesto que utiliza una combinación de protocolos en el túnel que proporciona seguridad en el paquete de la capa IP. Los principales protocolos son dos, el encabezado de autenticación (AH) que garantiza la integridad de los datos y la autenticación de origen, pero no la privacidad; y la carga útil de seguridad encapsulada (ESP) en el que se mantiene la integridad de datos, la autenticación de origen y la privacidad. Este protocolo opera en dos modos, el modo de transporte en donde la carga útil de la capa de transporte está encapsulada para garantizar la protección, y el modo de

túnel que no proporciona protección alguna para el encabezado IP, pero aquí todo el paquete IP, incluido el encabezado IP, está protegido en el modo de túnel IPSec.

L2TP

El protocolo de tunelización de capa 2, es un tipo de técnica de túnel VPN de acceso remoto. Las principales características de seguridad, como autenticación sólida, cifrado, confidencialidad e integridad, no se proporcionan por paquete, pero el túnel IPSec puede brindarle este tipo de seguridad. Por otra parte, el túnel IPSec por sí solo no puede crear un túnel para paquetes de Capa 2, por ello es necesario la combinación de estos dos protocolos para una seguridad óptima por paquete inclusive en la capa 2. Esta combinación encapsula el paquete más veces, por esto se garantiza más privacidad y seguridad en la transmisión.(Jahan et al., 2017).

SSL

Establece un enlace cifrado entre un servidor y un cliente. SSL protege la comunicación al proporcionar cifrado, integridad y autenticación de mensajes. Este estándar permite que los componentes del sistema negocien los mecanismos de cifrado, autenticación de cifrado, autenticación e integridad que se utilizarán. SSL utiliza tres algoritmos criptográficos interdependientes. La autenticación, que permite que el cliente identifique al servidor y, de manera opcional, permite que el servidor identifique al cliente; la autenticación se logra mediante el cifrado de clave pública y un certificado digital. La confidencialidad utiliza criptografía simétrica para intercambiar mensajes de forma confidencial. La integridad frente a interferencias se realiza mediante el resumen de mensajes. EL protocolo SSL utiliza tres protocolos para implementar los algoritmos anteriores: el protocolo de enlace, el protocolo de registro y el protocolo de alerta. (Zaw et al., 2019)

EAP-TLS

El protocolo de autenticación extensible (EAP) definido en el RFC 3748(Chen et al., 2020) brinda soporte para múltiples métodos de autenticación. El método de seguridad en la capa de transporte TLS, proporciona autenticación mutua, negociación de conjuntos de cifrado con integridad protegida e intercambio entre dos puntos finales.

El protocolo de autenticación EAP-TLS aplica el método EAP para cargar procedimientos TLS. Utiliza un certificado digital de infraestructura de clave pública (PKI) para el cliente servidor para proporcionar autenticación mutua entre ellos. El certificado PKI contiene la información del usuario o a su vez el nombre del servidor. Este protocolo genera y distribuye dinámicamente claves de cifrado basadas en sesión y en el usuario para así mantener la seguridad durante la comunicación, por lo cual se considera un protocolo muy seguro(Prakash & Kumar, 2018).

WPA

El método de cifrado WPA utiliza un protocolo de integridad de clave temporal (TKIP) para el cifrado de datos. Esto elimina el uso de la misma la clave al momento de cifrar y se genera una clave diferente al azar para cada paquete de datos, cada clave de cifrado es de 128 bits para cifrar el paquete de datos. Sin embargo, ningún procedimiento garantiza totalmente la seguridad en la transmisión de datos; un ataque de fuerza bruta puede violar la seguridad de este protocolo pues no utiliza algoritmos criptográficos robustos(Sari & Karay, 2015).

WPA2

Los fallos en la seguridad de los algoritmos criptográficos dan como resultado nuevos algoritmos más robustos que los anteriores, es el caso de WPA2, también conocido como IEEE 802.11i (Alblwi et al., 2017) que es el resultado de la falla de protocolos anteriores (WEP – WPA).

WPA2 inicia los fundamentos de cifrado y proporciona privacidad e integridad de los datos de la seguridad inalámbrica: dos términos que estaban ausentes en los protocolos de cifrado anteriores. WPA2 utilizó el algoritmo de cifrado AES.

WPA Enterprise

Las redes empresariales más grandes requieren un servidor dedicado para automatizar y administrar la autenticación y los acuerdos de claves. La integración de los puntos de acceso de la organización con un servidor de autenticación como RADIUS proporciona credenciales únicas para cada dispositivo o usuario asociado. Cuando se utiliza WPA2 Enterprise, cada usuario tiene su propio nombre de usuario y contraseña. La información de autenticación por dispositivo y las claves de cifrado se intercambian a través de EAP. EAP es un marco de autenticación flexible que admite múltiples métodos de autenticación y siempre se implementa como parte de la arquitectura 802.1x. El servidor de autenticación autentica a los clientes y exporta las claves criptográficas generadas aleatoriamente en caso de utilizar el método EAP de material de claves derivado. El servidor de autenticación AS se comunica con los clientes a través de autenticadores que actúan como dispositivo de paso. AS podría implementarse como un servicio en el AP. EAP se encapsula utilizando varios esquemas de autenticación como Message Digest 5 (MD5), Transport Layer Security (TLS), Tunneled TLS (TTLS) y Protected Extended Authentication Protocol (PEAP). (Abo-Soliman & Azer, 2018)

2.3 Criptografía

En un sentido etimológico la palabra criptografía proviene del griego ‘Kriptos’ que se traduce a ocultar y la palabra ‘Graphos’ escritura, con lo cual se da a entender que la idea principal es ocultar la escritura; en el contexto de la seguridad de la información, se trata de una técnica empleada para garantizar la integridad y la privacidad de la información transformando un mensaje legible en ininteligible al cifrarlo. (Granados, 2016)

Actualmente la criptografía se puede clasificar en Criptografía simétrica y criptografía asimétrica.

2.3.1 Criptografía Simétrica

Es un sistema que hace uso de una sola llave secreta para cifrar y descifrar un mensaje. Dicho de otro modo, se tiene un mensaje *m* y por alguna técnica de criptografía simétrica es cifrado con la clave secreta k dando como resultado un mensaje cifrado m', el cual se podrá descifrar únicamente con k por lo que cada debe ser compartida exclusivamente con el receptor.

Una de las principales ventajas de la criptografía simétrica es que puede garantizar confidencialidad dentro de una comunicación por el hecho de que la clave será compartida únicamente con quienes se desea que conozcan el contenido del mensaje.

Pero bajo estas condiciones si se tiene varios receptores es necesario generar una llave secreta para cada uno de ellos lo que resulta muy complicado para el emisor al momento descifrar el mensaje con cada una de las claves generadas y se presenta un problema en la estación emisora del mensaje para poder administrar todas las claves.

Otro inconveniente importante que se presenta en la criptografía simétrica es encontrar la forma de dar a conocer la clave privada al destino de manera segura, esto disminuye la garantía de seguridad de la información ya que si un tercero logra obtener la clave por consiguiente puede conocer el mensaje.(Granados, 2016)

2.3.1.1 AES. Este tipo de cifrado se realiza por bloques y es de los más comunes que se utilizan en diversas aplicaciones. Este algoritmo tiene una estructura particular propia para cifrar y descifrar datos sensibles y se aplica en hardware y software en todo el mundo. Es extremadamente difícil para los piratas informáticos obtener los datos reales al cifrar mediante el algoritmo AES. AES tiene la capacidad de manejar tres tamaños de clave diferentes, como AES 128, 192 y 256 bits y cada uno de estos cifrados tiene un tamaño de bloque de 128 bits. (Ako, 2017)

2.3.1.2 DES. Es un algoritmo de cifrado por bloques que es popular porque se utiliza como un algoritmo de cifrado de simetría de clave estándar. DES opera en un tamaño de bloque de 64 bits, encripta texto plano de 64 bits en texto cifrado de 64 bits utilizando 56 claves o subclaves internas. La clave interna se genera a partir de una clave externa (clave externa) de 64 bits de longitud. En el algoritmo de cifrado DES, la clave utilizada en el proceso de cifrado y descifrado debe ser la misma para que los datos puedan volver a su forma original. Podría ser, debido a esta "similitud", el DES también se denomina algoritmo de cifrado simétrico. La esencia del proceso de cifrado es ocultar los datos difuminando los datos "auténticos" y reduciendo la regularidad de la información para que los datos no puedan ser "leídos" excepto por la parte legítima. El algoritmo de cifrado DES funciona procesando bloques de datos de 8 bytes (64 bits) con bloques de claves de 8 bytes (64 bits). (Fernando et al., 2019)

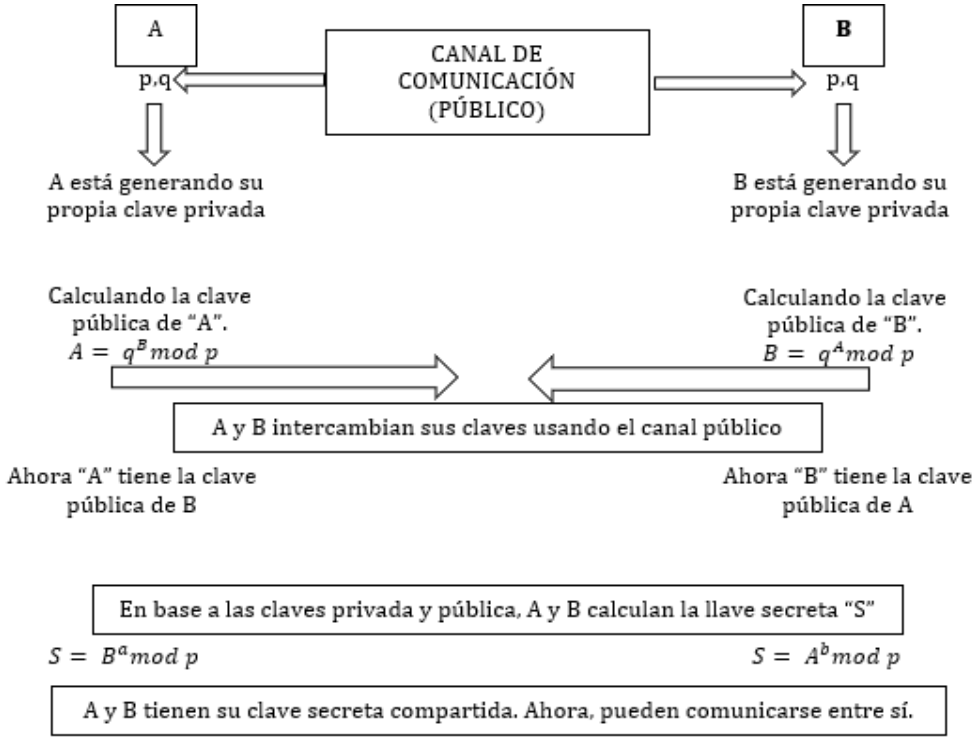
2.3.1.3 3-DES. Es un algoritmo de simetría de la criptografía moderna. En general, 3-DES se mejora con respecto al Estándar de cifrado de datos (DES) que todavía se usa comúnmente hasta el momento actual. El nivel de seguridad 3-DES mejor que DES se convierte en la principal razón para el uso del algoritmo 3-DES. El esquema del algoritmo 3-DES es muy similar al DES, pero en 3-DES se necesitan tres claves para cifrar y descifrar el mensaje.

2.3.2 Criptografía Asimétrica

Es el tipo de criptografía en el que se realiza la encriptación con llave pública y se descifra con llave privada o viceversa. Representa una gran ventaja porque reduce la cantidad de llaves a ser administradas. Algunos ejemplos de este tipo de criptografía son Diffie Hellman, RSA y Curvas Elípticas.

2.3.2.1 Algoritmo Diffie Hellman. Es un algoritmo de intercambio de claves que usa aritmética modular y logaritmo discreto para obtener una clave común para el emisor y el receptor quienes eligen un número primo común *p* y *q* como su raíz primitiva donde q < p.

Figura 6
Intercambio de claves Diffie-Hellman.



Tomado de "Enhanced diffie-hellman algorithm for reliable key exchange" por (Aryan et al., 2017).

En la Figura 6 se muestra el funcionamiento del algoritmo en donde A y B realizan un intercambio de claves en un canal de comunicación público. Luego de acordar los números primos p y q, ahora eligen su propia clave privada "a" y "b" respectivamente. Ahora para

intercambiar la clave, A calculará su clave pública que es $A = q \times a \times mod\ p$. y la intercambiará con la clave pública de B, que es $B = q \times b \times mod\ p$. Ahora para obtener la clave secreta compartida, A y B deberán calcular la clave privada y encontrarán que obtuvieron una clave secreta compartida, $S = B \times a \times mod\ p = A\ \times b \times mod\ p$. El algoritmo Diffie-Hellman proporciona una clave secreta compartida tomando la raíz primitiva del número primo y encontrando las claves públicas e intercambiándolas a través del canal público y utilizando aritmética modular. (Aryan et al., 2017)

Existen muchos ataques posibles en el algoritmo Diffie-Hellman, como ataque de texto plano conocido, ataque de intermediario, ataque interno, ataque externo, etc. En el ataque Man-in-the-middle, el atacante conserva las claves públicas tanto del emisor como del receptor y les envía claves públicas falsas.

2.3.2.2 Algoritmo RSA. Es una técnica criptográfica de clave pública usado para cifrado, intercambio de claves y firma digital. En este sistema cualquier persona puede cifrar los datos, pero solamente quien se autentique podrá descifrarlos.

Al aplicar algunos cálculos matemáticos de dos números primos grandes, se generan las claves. La seguridad del criptosistema RSA depende de las dificultades de la factorización de números primos grandes. Se puede generar la Clave Privada usando información de clave pública, que incluye n (multiplicación de números primos), el atacante no puede obtener el factor primo de *n* y por lo tanto no puede obtener la clave privada. Esto hace al algoritmo más seguro.

- Generación de claves RSA:
  1) Obtener dos números primos grandes p y q del mismo tamaño, relativamente, de modo que su producto *n= pxq* tenga una longitud de bits requerida, por ejemplo, 1024.
  2) Calcular n= pxq y $\varphi(n) = (p-1)(q-1)$

3) Elegir un cifrado entero aleatorio tal que mcd [e,φ(n)] = 1 y 1< e < φ(n).
4) Calcular el exponente secreto d en el rango 1 < d < φ de manera que: e x d = 1 mod φ(n).
5) La clave pública es (e,n) y la clave privada es (d,n).
   - Los valores secretos son d, p, q, y φ.
   - n se conoce como la multiplicación o módulo de los números primos.
   - e se conoce como exponente público o encriptación de exponente o simplemente exponente.
   - d se conoce como el exponente privado o el desencriptado del exponente.

- Cifrado RSA:

  El emisor realiza las siguientes operaciones:

  1) Determina la clave pública
  2) El mensaje de texto plano es representado como un número entero positivo.
  3) Calcula el texto cifrado: C = M . e . mod (n).
  4) Envía al receptor el texto cifrado.

- Descifrado RSA:

  El receptor hace lo siguiente:

  1) Utilice la clave privada (n,d) para calcular el texto plano M = C . d. mod (n)
  2) Extraiga el texto plano del mensaje representativo M.

2.3.2.3 Criptografía De Curva Elíptica. La criptografía de curva elíptica pertenece al grupo de la criptografía asimétrica, esto debido a que utiliza un par de claves tanto pública como privada, para cifrar y descifrar el mensaje mediante algún procedimiento de manera que, aunque un tercero conozca la clave pública, no pueda determinar la clave privada.

Para comprender este tipo de criptografía se inicia con la descripción de un campo finito sobre el cual se encuentra la curva elíptica.

Campos Finitos

Una curva elíptica sobre un campo K es un conjunto formado por el punto al infinito y los puntos P=(x,y) Є KxK que satisfacen la Ecuación 1 denominada Ecuación de Weistrass. (Karina & Antonio, 2012)

Ecuación 1
Ecuación de Weierstrass

$$E(K): y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

La Curva Elíptica E(K) junto con el punto al infinito (∞) forman el grupo aditivo {E(K) U ∞, +}, donde el punto ∞ es el elemento identidad (P+∞=P). Para aplicaciones criptográficas, el campo K es un campo finito. Si K es un campo primo GF(p) la ecuación de la curva elíptica se describe en la Ecuación 2:

Ecuación 2
Ecuación EC de un campo primo GF

$$E(GF_p): y^2 = x^3 + ax + b$$

Cuando la curva elíptica está definida en el campo binario GF_{2m}, la ecuación queda descrita como muestra la Ecuación 3:

Ecuación 3
Ecuación EC del campo binario GF

$$E(GF_{2m}): y^2 + xy = x^3 + ax^2 + b$$

Problema Del Logaritmo Discreto Elíptico

Dada una curva E en un campo finito, la operación principal de una curva elíptica denominada multiplicación escalar se puede representar de la siguiente manera:

Ecuación 4
Multiplicación escalar de una EC

$$Q = dP$$

Donde:

P y Q: son puntos de la curva elíptica

d: es un escalar discreto

El problema del Logaritmo Discreto Elíptico (PLDE) se define entonces como determinar el escalar *d* dados los puntos *P y Q*.

De lo anterior nace la seguridad de las curvas elípticas, en la dificultad para resolver dicho problema, pues resulta más complejo descifrar el escalar *d* en comparación con la factorización de RSA o con el logaritmo discreto de Diffie-Hellman. (Menezes et al., 2011)

Esquemas Criptográficos De ECC

Entre los principales métodos criptográficos de curva elíptica se pueden notar los siguientes:

- **Esquema de Diffie Hellman en ECC (ECDH)** Este esquema permite generar e intercambiar claves entre dos entidades. Para su implementación se puede realizar en cuatro partes, en las dos primeras se generan las claves tanto para el emisor como para el receptor utilizando un Algoritmo de producto punto. Luego de realizar el intercambio, los dos extremos realizan los cálculos para encriptar/desencriptar el mensaje utilizando también el producto punto. (Barrera & Whebe, 2020)

- **Esquema de cifrado integrado de curva elíptica (ECIES**) Es usado para cifrar datos utilizando ECDH para generar dos llaves simétricas, tanto para el cifrado y la autenticación. El texto plano se enmascara en los puntos de la curva elíptica, usa compresión de puntos para minimizar el uso de la memoria utilizada para almacenar puntos de la curva elíptica utilizada. En el proceso de cifrado, el usuario *A* cifra el mensaje *m* con la clave pública P_B del usuario *B* y genera el texto cifrado *(Q, C, t). h* es el cofactor que satisface $hn = \#E(F_p)$. *KDF(.)* es la función de derivación de claves basada en una función hash. $ENC_{k1}(.)$ y $DEC_{k1}(.)$ son generalmente funciones de algoritmos simétricos y también pueden ser operaciones XOR simples, donde k_1 es la clave secreta de *ENC* y *DEC*. $MAC_{k2}(.)$ es el código de autenticación de mensajes con la clave k_2, como HMAC. Hay dos operaciones de multiplicación escalar durante el cifrado, entre las cuales el resultado *Q* de *kG* se transmite a *B* como parte del texto cifrado. *B* usa su propia clave privada d_B para descifrar el texto cifrado enviado desde *A.* Para resistir un ataque, se requiere la validación del punto de entrada *Q* antes de la multiplicación escalar en el paso 1. En la Tabla 1 se muestra los procesos de cifrado y descifrado del Esquema ECIES, cuyos procesos se explican mediante diagramas en la Figura 7 y Figura 8.

Tabla 1

Algoritmos de Cifrado y Descifrado ECIES.

Algoritmo 1 Cifrado de ECIES

Requisito: La definición de una curva elíptica específica E(a,b) en F_p, un punto base G de la curva con orden n, texto plano m, clave pública de B, P_B. Se obtiene: *Texto cifrado (Q, C, t).*

1. Generar *k* de forma aleatoria (generador).
2. Calcular $Q = kG$ y $S = hkP_B$, donde $Q = (x_Q, y_Q)$ y
   S = (x_S, y_S);
3. Si S = 0 regresar al Paso 1.
4. Calcular $(k_1, k_2) = KDF\ (x_s, Q)$, $C = ENC_{k1}(m)$ y
   $t = MAC_{k2}(C)$;
5. Devuelve *(Q, C, t)*

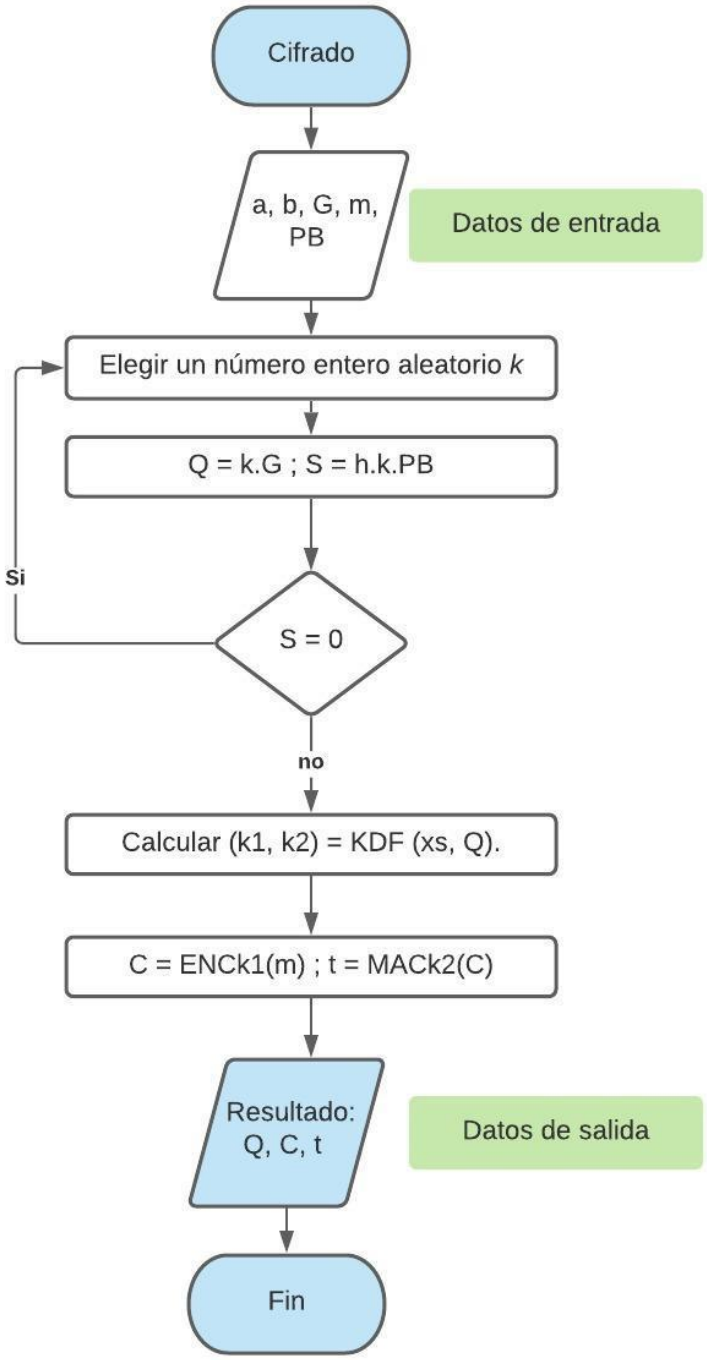
Algoritmo 2 Descifrado de ECIES

Requisito: La definición de una curva elíptica específica E(a,b) en F_p, un punto base G de la curva con orden n, texto plano m, clave privada de B, d_B y el texto cifrado C. Se obtiene: Texto plano *m.*

1. Comprobar el punto de entrada Q con validación. Si la validación falla, regresa rechazando el texto cifrado.
2. Calcular $S = (hd_B)Q$, *donde* $S = (x_S, y_S)$. *Si S = O,* regresa rechazando el texto cifrado.
3. Calcular $(k_1, k_2) = KDF\ (x_s, Q)$ y $t' = MAC_{k2}$(C). Si $t' = t$ regresa rechazando el texto cifrado.
4. Calcular $m = DEC_{k1}(C)$.
5. Devuelve *(m)*

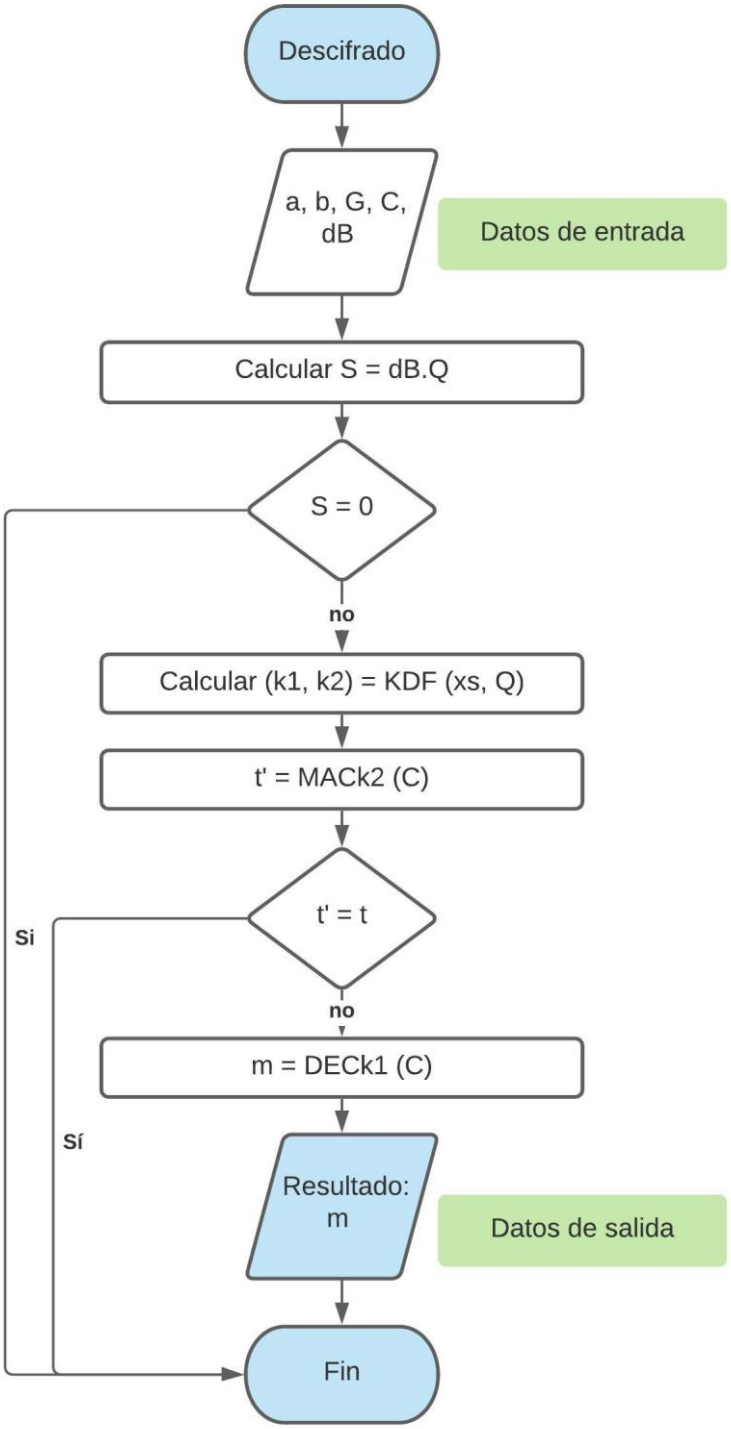
Tomado de "A new weak curve fault attack on ECIES: embedded point validation is not enough during decryption" por (Cao et al., 2021)

Figura 7
Diagrama de Flujo de Cifrado ECIES.



Recuperado de(Cao et al., 2021).

Figura 8
Diagrama de Flujo de Descifrado ECIES.



Recuperado de(Cao et al., 2021).

- **Algoritmo de firma digital de curva elíptica (ECDSA**) Puede dividirse en dos etapas, en primer lugar, se genera la firma digital y la segunda etapa es de verificación. En cada etapa se utiliza una llave privada y una función hash.

Jerarquía de operaciones en ECC

Los algoritmos de curvas elípticas trabajan con un modelo jerárquico, compuesto de algunas operaciones que se aplican en un campo finito de números, dando lugar a la jerarquía de operaciones por capas, según (Kaschel & Abarzua, 2018) se pueden notar cinco capas de operaciones en ECC:

- *Aritmética de campos finitos***:** Suma, Elevar al cuadrado, Traza, Multiplicación, Raíz cuadrada, Media Traza.
- *Aritmética de curvas elípticas:* Suma de puntos, doblado de puntos, bisección de un punto.
- *Primitivas de curvas elípticas:* Generación de llaves, Firma/verificación, Multiplicación escalar.
- *Protocolos de curvas elípticas:* Diffie-Hellman, Autenticación, Otros.
- *Aplicaciones***:** E-commerce, Monedero digital, comunicaciones seguras, otros.

2.4 Sistemas De Alerta Temprana

Los sistemas de alerta temprana tienen como objetivo principal el monitoreo de ciertos fenómenos naturales con el fin de brindar protección al medio ambiente o los asentamientos humanos cercanos a este fenómeno. Por lo tanto, es primordial es uso de diferentes tecnologías que permitan la gestión adecuada de las redes desplegadas para monitorear el comportamiento de dichos fenómenos.

2.4.1 Sistemas De Monitorización

El monitoreo es una herramienta qué puede cumplir diferentes perspectivas que generalmente están basadas en plataformas web, habitualmente son de sencilla arquitectura. permiten mostrar algunos parámetros como sensores, gráficas de datos, establecer condiciones sobre posibles alertas o cambios frente a límites establecidos y en algunos casos enviar notificaciones que podrían verse por cualquier persona o simplemente restringirse para que sea notificado al administrador.

2.4.1.1 Características De Un Sistema De Monitoreo. Entre las características principales de un sistema de monitoreo se pueden encontrar las siguientes:

- Es posible generar notificaciones qué alerten sobre eventos ocurridos en las variables que se están monitoreando
- Es posible dar valores máximos y mínimos en base a las variables que se monitorean
- Se puede generar gráficas interactivas
- Los servicios pueden funcionar 24 horas al día, si fuese necesario.
- Monitoreo del uso de una máquina o un dispositivo
- Datos en tiempo real

2.4.1.2 Ventajas De Monitorear. Un sistema de monitorización brinda ventajas de ordenamiento y calidad, a continuación, se detalla algunas de ellas

- En el caso de una empresa u organización hace posible mantener la credibilidad al contar con un sistema de monitoreo estable.
- Permite chequear servicios, dispositivos dañados o detenidos por algún problema que se esté presentando.

- En algunos casos es posible conectar o integrar una base de datos, por eso hay que notar que estos elementos suman cuando se trata de dar mejoras a una organización.
- Un sistema de monitoreo puede representar varias ventajas, ya que permite mantener operaciones organizadas, en tiempo real y funcionando de forma efectiva, con esto se puede aumentar la calidad de servicio.

2.5 ISO IEC IEEE 29148

Es un estándar que define conjunto de pasos que sirven como modelo en el análisis de los requerimientos del software y del hardware en función del ciclo de vida del desarrollo del sistema. Permite ser empleado como una propuesta de mejora en diferentes áreas de esta investigación, tales como: funcionalidades, limitaciones, tratamiento de procesos, guías, software, actividades y tareas.

2.5.1 Pasos de la norma IEEE 29148

Los pasos que se especifican en el estándar IEEE 29148 son: definición de los requisitos de las partes interesadas donde se identifica a los actores y sus requerimientos, análisis de requerimientos que inicia con la definición de requerimientos para su posterior análisis, actividades de ingeniería en otros procesos técnicos que permite demostrar el cumplimiento de los requisitos y la gestión de requisitos que permite mantener los requisitos. Un paso se define como un proceso conformado en ocasiones de subprocesos que son evaluados mediante herramientas, cuya intención es identificar las necesidades a combatir con el uso del sistema. La Tabla 2 muestra los procesos, subprocesos y herramientas especificadas por el estándar IEEE 29148.

Tabla 2

Pasos del estándar IEEE 29148.

Paso	Proceso		Subprocesos	Herramientas
1	Definición de los requisitos de las partes interesadas		Identificar los actores individuales o grupo. Obtener requisitos identificados por los actores	Observación directa e indirecta Entrevistas Cuestionarios Talleres estructurados con lluvias de ideas
2	Análisis de requerimientos.	2.1. Definir los requisitos del sistema	Definir la frontera funcional del prototipo Definir cada función que se requiera. Definir restricciones necesarias Definir técnicas y criterios de calidad que permitan la evaluación del ciclo de vida del prototipo Especificar los requisitos y funciones del prototipo	Definición de adentro y hacia afuera Comparación por pares y ranking Escala por niveles
		2.2. Analizar y Mantener los requisitos del sistema	Analizar la integridad de los requisitos del sistema Feed back de los requisitos con la parte interesada Demostrar la trazabilidad entre los requisitos del prototipo y los requisitos de las partes interesadas Mantener durante el ciclo de vida el conjunto de requisitos del prototipo, junto con los fundamentos asociados, decisiones y suposiciones	
3	Actividades de ingeniería en otros procesos técnicos Requisitos de validación y verificación		Plan sobre la base de los requisitos del sistema Ejecución del plan para demostrar el cumplimiento o conformidad de los requisitos de diseño específicos	Revisión de requisitos Prototipado Generación de casos de prueba Revisiones con stakeholders
4	Gestión de requisitos. Son tareas que registran y mantienen los requisitos		Gestión del cambio en adoptar medidas para mitigar Medición de los requerimientos para la calidad del producto Configuración, información, medición para el plan de gestión y ejecución	
	Elementos de información		Documentos de requerimientos del stakeholder (StRS) Documentos de requerimientos del sistema (SyRS) Documentos de especificaciones de requerimientos (SRS)	

2.6 Software de Simulación para entornos WSN

Existen varias opciones al momento de simular redes, en esta ocasión se realiza una investigación entre las más importantes y utilizadas para realizar finalmente una tabla

comparativa en la que se evidencia las diferencias y semejanzas que existen entre ellas, así poder evaluar y evidenciar cuál se adapta a las necesidades del presente proyecto.

2.6.1 OMNET

OMNeT++ es una biblioteca y un marco de simulación de C++ extensible, modular y basado en componentes, principalmente para crear simuladores de red. (OpenSim Ltd., 2022)

Aunque OMNeT++ no es un simulador de red en sí mismo, ha ganado una gran popularidad como plataforma de simulación de red en la comunidad científica, así como en entornos industriales, y ha creado una gran comunidad de usuarios.

OMNeT++ proporciona una arquitectura de componentes para modelos. Los componentes (módulos) se programan en C++, luego se ensamblan en componentes y modelos más grandes utilizando un lenguaje de alto nivel (NED). La reutilización de los modelos es gratuita. OMNeT++ tiene una amplia compatibilidad con GUI y, debido a su arquitectura modular, el kernel de simulación (y los modelos) se pueden integrar fácilmente en sus aplicaciones.

La biblioteca modelo de protocolo estándar de OMNeT++ es INET Framework (.INET), que contiene modelos para la pila de Internet y muchos otros protocolos y componentes. Sin embargo, existe una amplia lista de modelos que pueden ser usados en simulaciones con OMNeT++. El kernel de simulación es C++ estándar y se ejecuta básicamente en todas las plataformas en las que se encuentra disponible un compilador C++ moderno. El IDE de simulación requiere Windows, Linux o MacOS.

2.6.2 NS2

Es un simulador de redes opensource usado en ambientes de investigación que soporta los protocolos base de las redes de comunicación como son TCP, UDP, Routing, Wireless, Comunicaciones Satelitales, etc. Está desarrollado para el lenguaje C++ y la interfaz se maneja

en el lenguaje OTCL. Este simulador permite al diseñador crear o modificar nuevos protocolos, realizar mediciones de la red tales como jitter, throughput, estado de filas; además admite la caracterización de tráfico y todas las simulaciones que se programan pueden visualizarse de forma gráfica mediante NAM (Network Animator). Su última versión 2.35 fue lanzada el 4 de noviembre del 2011(University of Southern California, 2016). Está diseñado para sistemas operativos Linux, FreeBSD, Solaris, MacOSX y Windows (Cygwin)

2.6.3 NS3

El Network Simulator 3 (NS-3) es un simulador de redes para eventos discretos, es decir que cada evento ocurre en un instante determinado y tiene la capacidad de modificar el estado del sistema. Es usado en ambientes de investigación para educación por lo que es un simulador opensource; además, una ventaja es que permite construir un núcleo de simulación sólido, sencillo y bien documentado y en relación con su predecesor ns-2 permite extrar las trazas para su análisis. Funciona mejor en Sistemas operativos Linux (virtual o nativo). Otra característica que resalta de este simulador es que permite protocolos de comunicación más actuales sean basados en IP o no, por ejemplo, Wi-Fi, WiMAX, LTE, Redes de sensores, Redes TCP/IP, aplicaciones, etc. Está desarrollado para programarse en lenguaje C++ o Python y para compilar se recomienda utilizar ./waf.

2.6.4 MATLAB

Se trata de una plataforma de programación y cálculo numérico utilizada por millones de ingenieros y científicos para analizar datos, desarrollar algoritmos y crear modelos. (MathWorks, 2020). Posee un lenguaje de alto nivel el cual permite realizar cálculos científicos y de ingeniería, además cuenta con aplicaciones adicionales para realizar ajustes en el trazo de curvas, clasificación de datos, análisis de señales, etc. Para programar es compatible con

diversos lenguajes como C, C++, Java, .Net, Python, SQL, Hadoop e incluso aplicaciones como Excel de Microsoft.

2.6.5 OPNET (Optimized Network Engineering Tools)

OPNET Network Simulator, es una herramienta para simular el comportamiento y rendimiento de cualquier tipo de red. Se diferencia de otros simuladores debido a que es versátil. Gracias a IT Guru puede brindar modelos preconstruidos de protocolos y dispositivos. Se trata de un software libre de código abierto que ofrece gran cantidad de escenarios de proyectos realizando validación operativa, planificación y diseño de redes, modelado de protocolos y de tráfico de redes de telecomunicaciones, además, permite evaluar aspectos de rendimiento de sistemas de software complejos. La herramienta IT Gurú brinda facilidades para el diseño, debido a que se diseña en forma gráfica.

2.6.6 Resumen comparativo de Simuladores.

Existen varios tipos de software que permiten al desarrollador implementar redes con determinadas características de diseño, en la Tabla 3 se realiza una comparativa entre algunos de los más usados en simulación de redes de datos.

Tabla 3
Tabla comparativa entre simuladores de red

Características/Software de Simulación	*OMNET++*	*NS-2*	*NS-3*	*MATLAB*	*OPNET*
Lenguaje de Programación	C++, NED	C++ núcleo, Otcl interfaz	C++ núcleo, Python	M (Propio de MatLab), C y Fortran	Gráfico
Última Versión	6.0	2.35	3.35	R2022a	18.10.0
Tipo de licencia	Libre (para fines académicos)	Libre	Libre	Software privativo	Libre (Académico), Usuario Único (Modeler).
Licencia	APL de GPL	GNU de GPL versión 2	GNU de GPL	MathWorks	Riverbed Modeler (Windows/Linux)

Sistema Operativo	GNU/Linux y Windows	GNU/Linux, SunOS, Solaris, Cygwin, POSIX (FreeBSD)	POSIX (GNU/Linux, BSD), OS X y Windows (Cygwin / MinGW)	Windows, iOS y Linux	Windows
Código Público	SI APLICA	SI APLICA	SI APLICA	NO APLICA	NO APLICA
Capas del modelo OSI en las que trabajan	TODAS	TODAS	TODAS	TODAS	Enlace de datos, Red, Aplicación
Tipos de redes que se pueden simular	DATOS, WIRED, WIRELESS	DATOS, WIRED, WIRELESS	DATOS, WIRED, WIRELESS	DATOS, WIRED, WIRELESS, CIRCUITOS, MODELACIÓN.	DATOS, WIRED, WIRELESS
Uso Investigativo	ALTO	ALTO	MEDIO	ALTO	ALTO

2.7 Trabajos Relacionados

En (Louw et al., 2016) se analiza la magnitud del desafío que implica asegurar las WSN principalmente por las limitaciones de recursos con que cuentan este tipo de redes. También se propone el uso de criptografía de curva elíptica como mejor solución para alcanzar el mismo nivel de seguridad que con los algoritmos PKC (Public Key) sin comprometer los recursos de la red ni de los dispositivos. De esta manera los autores diseñan un protocolo de distribución de claves utilizando funciones criptográficas basadas en ECC garantizando que el sistema es seguro y eficiente. Se utiliza el Sistema TinyECC el cual es una biblioteca de código abierto en el que se incluyen tres esquemas de ECC: ECDSA (Firma Digital), ECDH (Diffie-Hellman EC) y ECIES (Esquema de cifrado integrado). Este sistema se caracteriza por ser portátil, eficiente, configurable y consciente de las limitaciones de los dispositivos. Para la distribución de claves una vez que el usuario autoriza interactuar con una mota (segura) la aplicación de gestión. A continuación, envía la clave del sistema que cifra utilizando el esquema ECIES con la clave pública del cliente. La clave cifrada se protege con AES CCM utilizando la clave inicial para garantizar que llegue al receptor sin alteraciones. El cliente recibe y descifra el mensaje, descifra la clave con ECIES y con su propia clave privada, luego de esto establece la

clave del algoritmo AES en la clave del sistema. En cuanto a la privacidad y autenticación, si una mota ha sido autorizada y ha recibido la clave del sistema puede comenzar a recopilar datos. Cuando se desea transmitir estos datos de forma segura de nuevo al bridge, debe cifrar los datos agregando un MAC, con esto se garantiza que el bridge pueda verificar la autenticidad del paquete de información que recibe, aquí se debe implementar protección contra ataques de repetición para lo cual se puede agregar simplemente un contador secuencial no repetitivo en el paquete de datos.

En (Montoya, 2019) se realiza un estudio en primer lugar para identificar bajo qué tecnología inalámbrica se puede desarrollar un sistema IoT utilizando sensores Wearable, además se realiza un segundo estudio para determinar el mejor sistema de encriptación a utilizarse en un sistema de este tipo. En conclusión, se determina que la tecnología idónea es NFC, ya que se adapta muy bien a las limitantes que tienen los dispositivos IoT y como algoritmo de cifrado se concluye que el DHEC (Diffie – Hellman Elliptic Curve) proporciona mayor seguridad y mejor escalabilidad. Para determinar el algoritmo de cifrado, se basa en algunas variables determinadas como limitantes propias en IoT, el nivel de riesgo, consumo de procesamiento, consumo de energía, tiempo de ejecución, consumo de ancho de banda. Mediante un software desarrollado, se realiza la comprobación del funcionamiento de los algoritmos evaluados (AES, RSA, DHEC, HJP), en dicha aplicación se muestra la cantidad de iteraciones que cada uno realiza para cifrar el mensaje, de ahí se puede comparar el tiempo de envío, el consumo de energía y el consumo promedio de procesamiento.

En (Ravi et al., 2019) se utiliza una solución eficaz en términos de seguridad y velocidad de los nodos, esta solución se basa en dispositivos FPGA (Field Programmable Gate Arrays) que es reprogramable y configurable a cualquier tipo de situación según los requisitos, esto lo hace muy adecuado para implementaciones WSN pues en la programación se inicia por programar a nivel placa para luego integrarla a la red. En tema de seguridad no es recomendable

cualquier tipo de criptografía en WSN, aquí se realiza una encuesta que determina que tanto para WSN como para FPGA el algoritmo criptográfico recomendado es la Criptografía de curvas Elípticas. Las WSN tienen restricciones como área limitada, potencia y memoria, por las cuales quedan restringidos la mayoría de los algoritmos criptográficos convencionales. En esta solución primero se sugiere configurar FPGA, y luego implementarlo en la WSN, como ya se ha mencionado anteriormente. Este método puede servir como una mejor alternativa. La implementación de FPGA se utiliza para implementar algún tipo específico de aplicación WSN para controlar algoritmos de cifrado de capacidades de mayor potencia computacional, monitoreo ambiental. Además, el nodo sensor propuesto basado en FPGA abre una oportunidad para una nueva clase de aplicaciones, como aplicaciones WSN en tiempo real. Algunos autores presentan resultados obtenidos en placas Vertex basadas en FPGA, con frecuencias de reloj variables donde ECC se implementa en las placas Vertex basadas en FPGA. El mejor resultado muestra 7,72 µs para la multiplicación de puntos en ECC. El ECC propuesto en el trabajo mencionado muestra una mejora del 34 % en velocidad en comparación con el resultado anteriormente expuesto. En conclusión, se sugiere agregar ECC a FPGA para aumentar la seguridad, ya que ECC soluciona problemas como el tiempo de cómputo limitado, batería limitada, espacio de memoria pequeño, etc. Según la aplicación se determina una clave eficiente para minimizar estos problemas. El tiempo de cálculo se redujo a 3,5 µs, debido en gran parte a que se realiza un cálculo paralelo gracias a FPGA, donde el nodo está recopilando o procesando datos mientras calcula la clave. Por lo tanto, hace que el nodo WSN sea de naturaleza más eficiente y dinámica.

Gracias a estos trabajos previos, se puede determinar que existen diversidad de aplicaciones de Criptografía de Curva Elíptica y todos permiten una mayor eficiencia del sistema en cuanto a seguridad. Además, coinciden en que este tipo de criptografía es idóneo para implementaciones IoT, específicamente WSN, ya que estos sistemas poseen grandes

limitaciones con las cuales los diferentes esquemas y formas de aplicación de las ECC no generan ningún inconveniente, por el contrario, usa métodos alternativos que permiten optimizar los recursos.

CAPÍTULO III

Diseño de la solución

En este capítulo se aplica la metodología para encontrar la solución al problema, con el fin de realizar un trabajo estructurado y determinar de manera óptima los requerimientos para la implementación del algoritmo de seguridad según lo que se ha establecido en el alcance del presente trabajo, para esto se inicia describiendo la metodología que se aplica.

3.1 Metodología

Para la implementación del algoritmo de seguridad en una red de sensores simulada, se requiere utilizar técnicas de recolección de información necesaria para establecer los requerimientos que permitan llegar a una solución óptima y la respectiva verificación de estos. Por esta razón se toma en cuenta los procesos sugeridos por la norma IEEE 29148 en el establecimiento adecuado de requerimientos. Esta normativa se considera la más completa siendo así la más utilizada en el desarrollo de sistemas de software.

3.2 Estándar ISO IEC IEEE 29148

Para diseñar un sistema relacionado con software es de vital importancia utilizar una guía de referencia que contenga un conjunto de técnicas y procesos para que la solución que se obtenga pueda ser verificada de manera objetiva y así cumpla con los requerimientos establecidos. La ingeniería de requisitos que presenta la normativa ISO IEC IEEE 29148 resulta aplicable al sistema que se quiere desarrollar, ya que posterior a la implementación existe un proceso de verificación en el que se busca cumplir con los requerimientos propuestos al inicio.

Esta norma sigue un proceso de definición de requerimientos en cuatro pasos, en el primero de ellos se define los requisitos de las partes interesadas, en el segundo se realiza un análisis de los requerimientos, en el tercero se determina las actividades de ingeniería que

satisfacen a los requerimientos seguidas de un proceso de verificación y por último realiza gestión de requisitos. Además, en el estándar descrito se hace uso de elementos de información como son: documentos de requerimientos de stakeholders (StRS), documentos de requerimientos de sistemas (SyRS) y documentos de especificaciones de requerimientos (SRS).

En el primer paso, para definir los requisitos de las partes interesadas, se siguen dos subprocesos: Identificar los actores individuales o grupo y obtener los requisitos identificados por los actores. Para lo cual se hará uso de herramientas como observación directa, y ficha técnica

Para el segundo paso que es el análisis de requerimientos en primer lugar se define los requisitos del sistema, para esto es necesario definir la frontera funcional del sistema detallando cada función que se requiera con sus respectivas restricciones, es decir, se especifica los requisitos y funciones de la solución. Luego se puede realizar procedimientos para analizar y mantener los requisitos del sistema como análisis de la integridad de los requisitos del sistema, feedback de requisitos con la parte interesada.

Como tercer paso se realiza actividades de ingeniería en procesos técnicos como validación y certificación. Se realiza un plan basado en los requisitos del sistema y se ejecuta el plan para mostrar conformidad con los requisitos de diseño establecidos generando casos de prueba con el fin de comparar entre los resultados y determinar si se ha dado cumplimiento a los requerimientos.

Finalmente se realiza la gestión de requisitos que son tareas para registrar y mantener los requisitos. En el caso de que existan errores se registran las medidas a adoptar para mitigarlos. En la medida de lo posible también se realizará una medición de requerimientos para mejorar la calidad del producto final.

3.3 Análisis

Para iniciar con el diseño de la solución al problema planteado, es necesario analizar todos los factores que intervienen en él, por ello se toma como base una propuesta de red de sensores inalámbricos diseñada para sistemas de alerta temprana de incendios con el fin de tomar los datos necesarios para poder realizar la simulación de una red, de esta manera los resultados puedan ser lo más cercanos a la realidad posible.

3.3.1 Situación Actual

Para realizar la implementación del sistema de seguridad basado en un algoritmo ECC en un ambiente de simulación, se ha tomado como base una red WSN que se encuentra propuesta en el trabajo investigativo "DISEÑO DE UNA RED INALÁMBRICA PARA UNA WSN DE UN SISTEMA DE ALERTA TEMPRANA DE INCENDIOS PARA EL "BOSQUE PROTECTOR GUAYABILLAS" (Enríquez & Michilena, 2018).

Los bosques son una fuente de recursos naturales, por lo que actualmente mediante los gobiernos seccionales y algunas organizaciones no gubernamentales se trata de mantenerlos libres de cualquier peligro que amenace a su ecosistema. De ahí la importancia de realizar la prueba en entornos simulados del uso de un algoritmo de seguridad en redes WSN de Alerta Temprana que permita demostrar que al implementarlo en la red no se evidencia variaciones en el performance, brindándole así una propuesta de solución viable para el intercambio seguro de datos y mantener a la vez las características y requerimientos de una red con recursos limitados.

3.3.2 Propósito y ámbito del sistema

La solución propuesta se basa en la implementación de un algoritmo criptográfico de bajo costo como son los algoritmos de Curvas Elípticas, sobre una red de sensores de alerta temprana diseñada para el bosque de Guayabillas. El objetivo principal es comprobar que un

algoritmo de la familia ECC brinda un nivel de seguridad robusto con tamaños de clave pequeños y pueden ser de mucha ayuda en este tipo de redes ya que no alteran significativamente el throughput de la red.

Todo esto se busca realizar en un ambiente de simulación que permita evidenciar el comportamiento de la red sin aplicar criptografía y comparar el valor de los principales parámetros con las características de la red luego de aplicar el algoritmo.

3.3.3 Descripción General

La solución consta de tres etapas que son: simulación de la red WSN, implementación del algoritmo de seguridad sobre la red y realización de pruebas midiendo diversos parámetros de la red. El algoritmo deberá garantizar la seguridad de la red y bajo costo computacional.

3.3.3.1 Funciones del Proyecto

Las funciones principales del proyecto son:

- Programar y configurar en ambiente de simulación una red WSN ya diseñada para el Bosque Protector Guayabillas, con 15 nodos sensores y un nodo central.
- Monitorear el comportamiento de la red WSN simulada ya diseñada para el Bosque Protector Guayabillas mediante la medición de parámetros básicos como tiempo de respuesta, costo computacional, ciclos por reloj, instrucciones por segundo, etc. En un ambiente de simulación antes y después de haber implementado el algoritmo criptográfico.

3.3.4 Parámetros de Diseño

La solución se basa en la implementación de un algoritmo criptográfico en una red WSN ya diseñada, los parámetros a tomarse en cuenta para realizar el proyecto son:

- Longitud de clave. - Pequeña

- Número de vueltas del algoritmo. - menor con respecto a RSA
- Librerías. - Para el monitoreo y la implementación del algoritmo.
- Costo Computacional. –Debe ser bajo debido a que se trata de una WSN y por ende no debe verse afectado en gran medida con el cifrado.

3.4 Requerimientos

Para desarrollar este proyecto es imprescindible que el algoritmo ECC se ajuste a las condiciones de la red simulada, por ello en primer lugar se inicia por implementar la red con las características de referencia para simular el escenario lo más real posible.

Una vez realizado el análisis en la sección anterior se puede determinar los requerimientos necesarios para el diseño de la solución. En primer lugar, se determina los actores directos e indirectos de este trabajo, para luego continuar con la determinación de requerimientos de Stakeholders y de sistema.

3.4.1 Stakeholders

Para recolectar información necesaria y determinar los requerimientos es necesario establecer en primer lugar los actores que están inmersos en el desarrollo del presente proyecto mediante el análisis y observación del entorno, así se obtiene en la Tabla 4 la lista de Stakeholders.

Tabla 4
Lista de Stakeholders

1.	Universidad Técnica del Norte
2.	Srta. Andrea Ortega
3.	MSc. Fabián Cuzme
4.	MSc. Jaime Michilena

3.4.2 Construcción y Atributos

Se debe plantear los requerimientos teniendo en cuenta a los actores que intervienen y las características de la solución que conducen a cumplir de mejor manera con todo lo requerido.

3.4.3 Nomenclatura para identificar los requerimientos

Para distinguir los requerimientos unos de otros se utiliza una nomenclatura determinada de manera que sea más fácil citarlos durante el desarrollo, implementación y pruebas. La nomenclatura para utilizarse se describe en la Tabla 5.

Tabla 5
Nomenclatura para Requerimientos

Requerimiento	Nomenclatura
De Stakeholders	*StSR*
De Sistema	*SySR*
De Arquitectura	*SrSH*

3.4.4 Requerimientos de Stakeholders

Una vez definidos los interesados en el desarrollo del proyecto y establecida la nomenclatura se determina los requerimientos usando técnicas de recolección de información como la lluvia de ideas, en este caso se ha realizado una ficha técnica de requerimientos, descrita como Anexo 1 en la sección de Anexos, en la cual el docente director del Proyecto y el Autor de este, declaran, en calidad de actores, los requerimientos que un sistema de seguridad para WSN en entorno de simulación necesita. Así, en resumen, los requerimientos determinados por los actores se describen en la Tabla 6:

Tabla 6
Requerimientos de Stakeholders

No.	*Requerimiento*	*Prioridad*		
		Alta	Media	Baja
StSR 1	Estabilidad de la red simulada	X		
StSR 2	Conocimiento en el lenguaje de programación	X		
StSR 3	Conocimiento en criptografía ECC	X		
StSR 4	Encriptación - Desencriptación	X		
StSR 5	Dispositivo Base de alto rendimiento		X	
StSR 6	Red WSN base	X		

3.4.5 Requerimientos del Sistema

Los requerimientos del sistema, al igual que los de Stakeholders, se determinan mediante el análisis de la funcionalidad de la solución, qué se requiere para cumplir con el objetivo del proyecto, y qué características de uso tendrá el sistema, por ello en la Tabla 7 se enumeran las funciones que debe realizar el sistema.

Tabla 7
Requerimientos del Sistema

No.	*Requerimiento*	*Prioridad*		
		Alta	Media	Baja
SySR 1	Red WSN en simulación	X		
SySR 2	Medición de dato Temperatura	X		
SySR 3	Medición de dato Humedad	X		
SySR 4	Tratamiento de información	X		
SySR 5	Monitoreo de parámetros	X		
SySR 6	Simplicidad del Lenguaje de Programación	X		
SySR 7	Manuales de instalación y manejo		X	
SySR 8	Menor tiempo de respuesta		X	
SySR 9	Seguridad	X		
SySR 10	Longitud de clave	X		

3.5 Recursos

Se especifica dos tipos de recursos, en un inicio se determina los recursos humanos requeridos para llevar a cabo la solución tomando en cuenta a las personas que se encuentran involucradas en la solución presentada, estás personas guían la investigación, aportan con ideas e intervienen en la toma de decisiones respecto al proyecto, y, posteriormente, los recursos materiales que se usan para realizar la investigación, todos los recursos se describen en la Tabla 8.

Tabla 8
Tabla de recursos

RECURSOS HUMANOS	
DESARROLLADOR	Srta. Andrea Ortega
TUTOR	MSc. Fabián Cuzme
ASESOR	MSc. Jaime Michilena
RECURSOS MATERIALES	
Computador	
Software de simulación	

3.6 Características de la red WSN

La red WSN de Alerta Temprana diseñada en (Enríquez & Michilena, 2018) para el Bosque de Guayabillas consta de 15 nodos sensores y un nodo Central cuya ubicación específica e interconexión se describen a continuación.

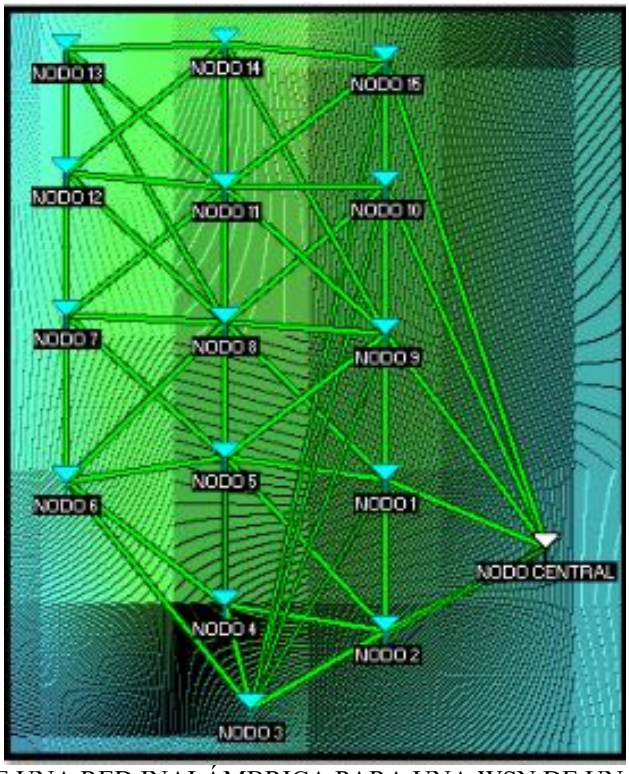
3.6.1 Topología de la Red

La red está interconectada con una topología en malla, de manera que se pueda garantizar la comunicación con vías de redundancia para que todos puedan llegar al Nodo Central. La tecnología de comunicación entre los nodos y el nodo central es XBee. En la Figura 9 se muestra la topología.

3.6.2 Ubicación de los nodos

El trabajo realizado por (Enríquez & Michilena, 2018) tiene una ubicación geográfica exacta de cada nodo basándose en la geolocalización de la plataforma Google Maps. En la Tabla 9 se muestra la información de los nodos.

Figura 9
Ubicación de nodos sensores en la red tomada como referencia



Tomado de "DISEÑO DE UNA RED INALÁMBRICA PARA UNA WSN DE UN SISTEMA DE ALERTA TEMPRANA DE INCENDIOS PARA EL BOSQUE PROTECTOR GUAYABILLAS " por (Enríquez Burgos, 2018)

Tabla 9
Ubicación geográfica de los nodos

Nodos	Latitud	Longitud
Nodo Central	0.336190	-78.106.107
Nodo 1	0.336625	-78.107.107
Nodo 2	0.335679	-78.107.158
Nodo 3	0.335190	-78.107.785
Nodo 4	0.335844	-78.108.370
Nodo 5	0.336755	-78.108.251
Nodo 6	0.336615	-78.109.140
Nodo 7	0.337646	-78.109.101
Nodo 8	0.337603	-78.108.204
Nodo 9	0.337534	-78.107.219
Nodo 10	0.338458	-78.107.272
Nodo 11	0.338437	-78.108.139
Nodo 12	0.338536	-78.109.062
Nodo 13	0.339304	-78.109.014
Nodo 14	0.339338	-78.108.143
Nodo 15	0.339242	-78.107.219

Tomado de "DISEÑO DE UNA RED INALÁMBRICA PARA UNA WSN DE UN SISTEMA DE ALERTA TEMPRANA DE INCENDIOS PARA EL BOSQUE PROTECTOR GUAYABILLAS " por (Enríquez Burgos, 2018)

3.7 Software utilizado

En la Tabla 3 se muestra una comparación entre los simuladores de red más utilizados, de ella se puede concluir que para efecto de una simulación WSN como la que se presenta en este proyecto, el Software idóneo es OMNet++, ya que éste permite trabajar en todas las capas del modelo OSI, además el lenguaje de programación C++ es conocido y fácil de manejar. Es un software de licencia libre, pero tiene uso investigativo alto con relación a los demás simuladores citados.

3.8 Diseño de la Solución

Tomando como base la comparación del algoritmo ECC con el algoritmo RSA en (Ravi et al., 2019) se obtiene en la Tabla 10 la relación de costos de cómputo y de performance de la red en un ambiente de simulación:

Tabla 10
Tabla Comparativa Costos y Performance RSA:ECC

COSTOS RELATIVOS DE CÓMPUTO			
Nivel de Seguridad (bits)			Relación Costo (RSA:ECC)
80			3:1
112			6:1
128			10:1
192			32:1
256			64:1
COMPARACIÓN DE PERFORMANCE DE RSA Y ECC			
	t(ms)	Operación/s	Ratio tamaño clave
ECC-160	3.69	271.3	1:6.4
RSA-1024	8.75	114.3	
ECC-224	5.12	195.5	1:9.1
RSA-2048	56.18	17.8	

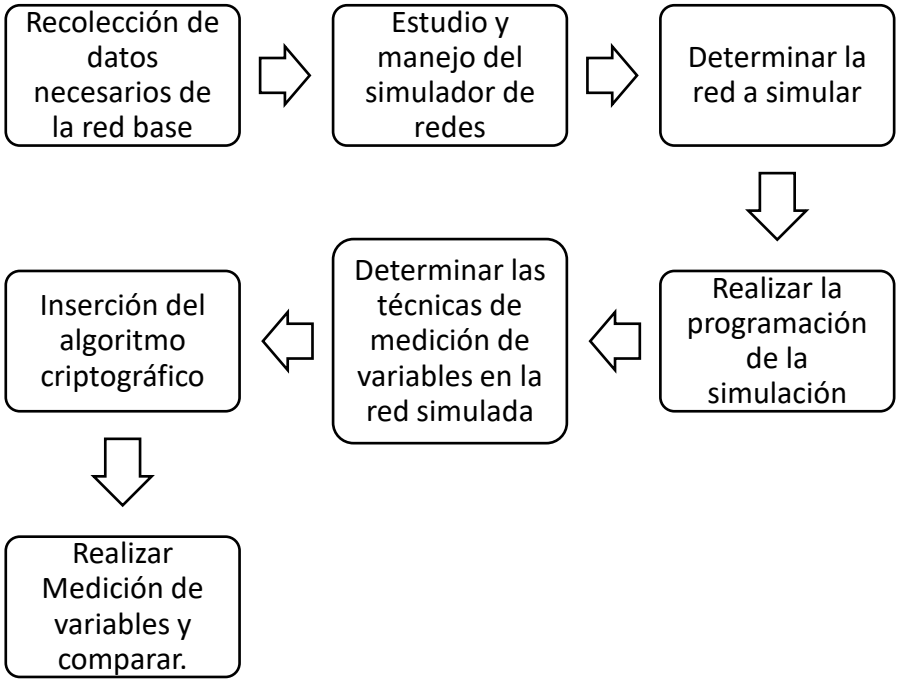
En lo que respecta a los costos de cómputo, según el nivel de Seguridad en bits, la relación Costo entre RSA y ECC sigue aumentando en lo que respecta a RSA mientras mayor sea el nivel de seguridad, lo que da una idea del bajo costo computacional que implica la implementación de algoritmos ECC.

En cuanto al performance de la red la criptografía de Curva Elíptica presenta menor tiempo de ejecución de operaciones en comparación con RSA. De igual manera al tener menor costo computacional en menor tiempo se concluye que ECC realiza mayor cantidad de operaciones por segundo. La criptografía de Curva Elíptica utiliza mayor longitud de clave para ofrecer el mismo nivel de robustez que RSA

3.8.1 Diagrama de bloques

Para una mejor comprensión del desarrollo de la investigación se ha elaborado el Diagrama de bloques de la Figura 10, especificando paso a paso cómo se pretende llegar a la solución.

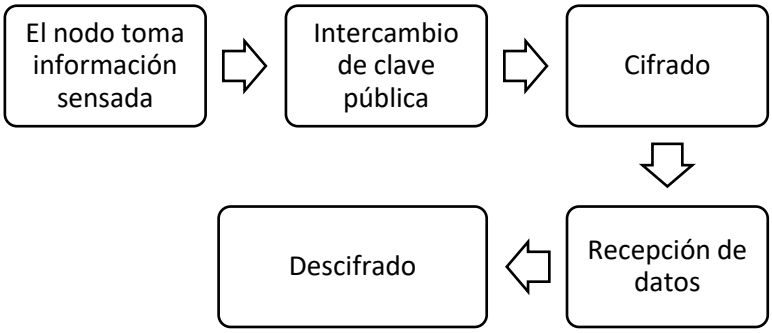
Figura 10
Diagrama de Bloques de la solución



Generalmente un nodo toma información y la encripta en este caso utilizando un algoritmo de la familia de curvas elípticas, luego utiliza el estándar 802.15.4 para transmitirla. Los algoritmos ECC brindan un nivel de seguridad similar al de otros algoritmos más utilizados como RSA, la diferencia es que al hacerlo consume menos recursos, por ello es recomendado

para aplicarlo en redes que requieran bajo costo, entre ellas las WSN. En la Figura 11 se representa la secuencia de pasos a seguir desde el nodo hasta el Gateway para la encriptación.

Figura 11
Diagrama de bloques de cifrado y descifrado



La solución funciona de la siguiente manera: en primer lugar, se realiza la autenticación con ECDSA, luego un intercambio de clave pública utilizando el algoritmo ECDH y finalmente se realiza la encriptación con ECDLP. Para los tres algoritmos se realiza operaciones de curva elíptica.

3.8.2 Diagramas de flujo de la función de los nodos

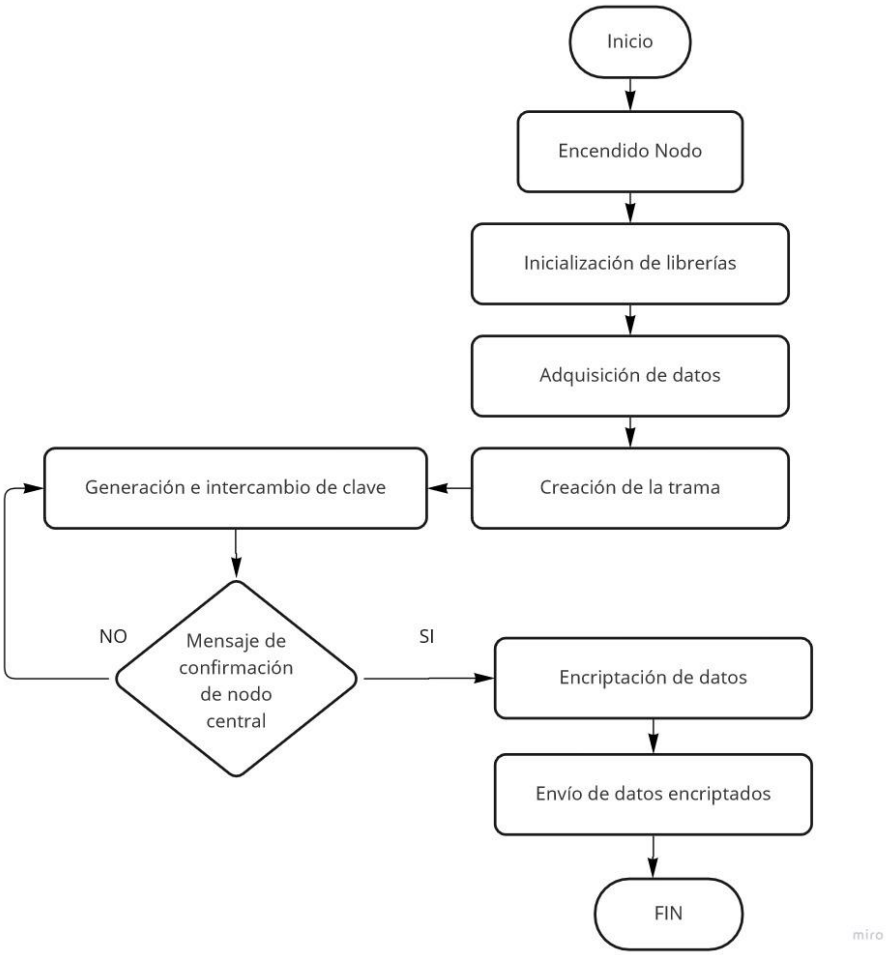
Para cada nodo sensor se configura un canal de comunicación dirigido al nodo central (Gateway). El nodo central solicita a cada nodo sensor recopilar la información, encriptarla y enviarla al nodo central, esto lo hace siguiendo un orden secuencial enviando la solicitud primero al nodo 1 y finalizando el ciclo con el nodo 15. Todos los clientes tienen la misma configuración, salvo que se diferencian en una distancia de cada nodo al Gateway para que dependiendo de esa distancia cada nodo transmita la información en determinados períodos de tiempo.

Al momento de asegurar los datos, los factores de la ecuación de ECC que se aplica, es calcular junto con los datos determinados por el valor del sensado de los nodos, al ser un ambiente de simulación estos datos se generan de forma aleatoria, de esta manera los datos se

aseguran, ya que si un nodo no tiene comunicación con el Gateway no podrá descifrar la información enviada por los demás nodos. La encriptación se realiza por medio de una clave pública compartida entre el sensor y el Gateway, haciendo uso de la clave privada de cada uno calculan un valor secreto que solamente el otro puede descifrar.

Figura 12

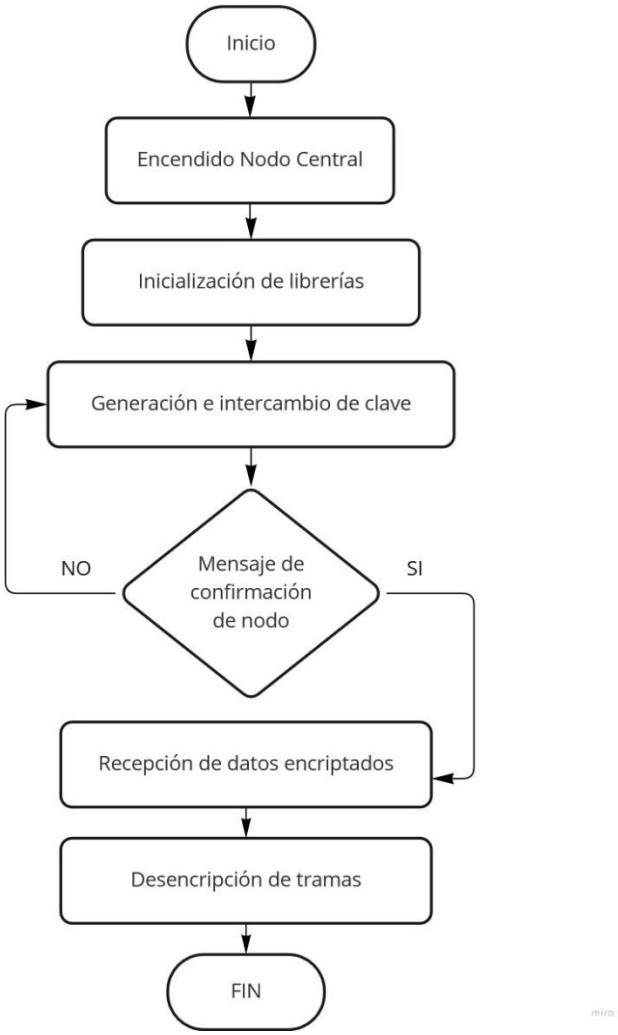
Diagrama de Flujo de la solución en el nodo sensor



Para calcular la clave se toma en cuenta la geometría de puntos basada en Curvas Elípticas, se refiere a los puntos donde la curva cambiaría de concavidad, para esto se debe realizar varias pruebas pues según la aplicación la ecuación requiere ajustes en la configuración. Inicialmente, el nodo sensor y el Gateway están de acuerdo en una curva particular con el punto base P. Generan sus claves públicas multiplicando P con sus claves privadas, a saber, K_S y K_G. Después de compartir claves públicas, generan una clave secreta

compartida multiplicando las claves públicas por sus claves privadas. La clave secreta es $R = K_S * Q_G = K_G * Q_S$. Con los valores conocidos de Q_S, Q_G y P, es computacionalmente intratable para un intruso calcular K_S y K_G, que son las claves privadas del nodo sensor y el Gateway. Como resultado, los adversarios no pueden averiguar cuál es la clave secreta compartida. El Gateway realiza este proceso para cada sensor.

Figura 13
Diagrama de Flujo de la solución en el nodo central



Durante la simulación el sistema muestra reportes en términos básicos, por lo que se hace uso de otro tipo de software que brinde las facilidades necesarias de monitorear la red específicamente en cuanto a la carga computacional que ejerce el algoritmo a la red.

Para comprender la secuencia lógica de pasos a seguir en el desarrollo de la solución se presenta a continuación en la Figura 18 y Figura 19 los Diagramas de flujo del nodo sensor y del nodo central respectivamente.

3.8.3 Diagrama de Secuencia del funcionamiento del algoritmo

En la Figura 14 se explica el proceso de Generación e Intercambio de claves haciendo uso del algoritmo ECDH. Posteriormente, en la Figura 15 se muestra el proceso de cifrado y descifrado que realizan los nodos. Los parámetros que necesita cada nodo para generar sus claves y para cifrar se describen en la Tabla 11.

Tabla 11
Descripción de los parámetros usados en el diagrama de secuencia.

Parámetro	Descripción
a y b	Coeficientes de la Ecuación
G	Punto Generador
r	Orden de G
p	Número primo que indica el tamaño del campo finito
E	Excentricidad de la curva
i_NC *i_n-1* *i_n*	Número entero aleatorio seleccionado por cada sensor
P_NC *P_n-1* *P_n*	Clave pública de cada sensor.
D y F	Puntos en común donde “x” de D o F es la contraseña común.
m	Mensaje por cifrar
S	Datos cifrados

Los diagramas descritos representan la secuencia lógica de pasos que realizan el Nodo Central, el nodo sensor más distante del nodo central y el nodo anterior al mismo.

Figura 14
Diagrama de Secuencia de Intercambio de claves con ECDH

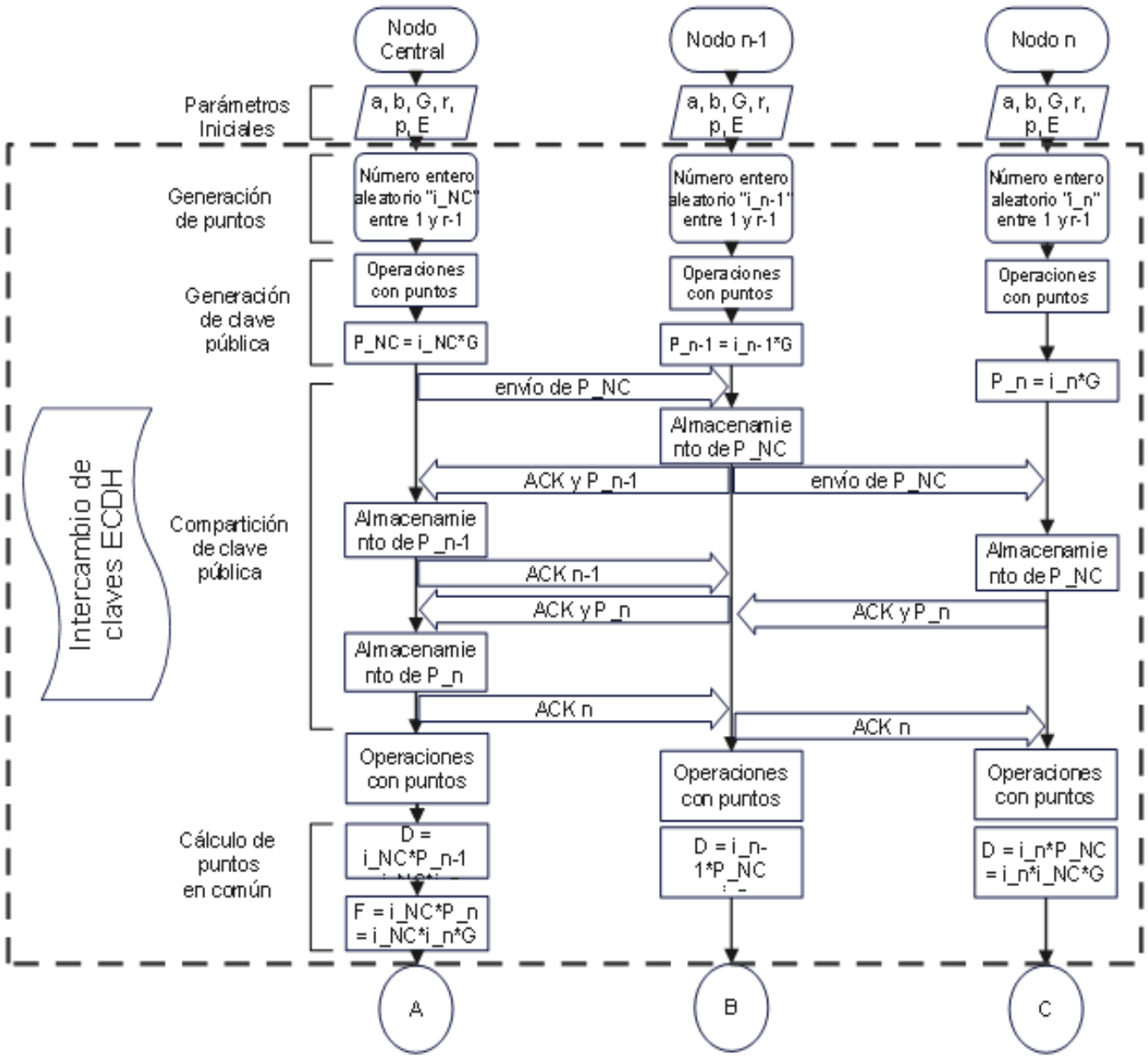
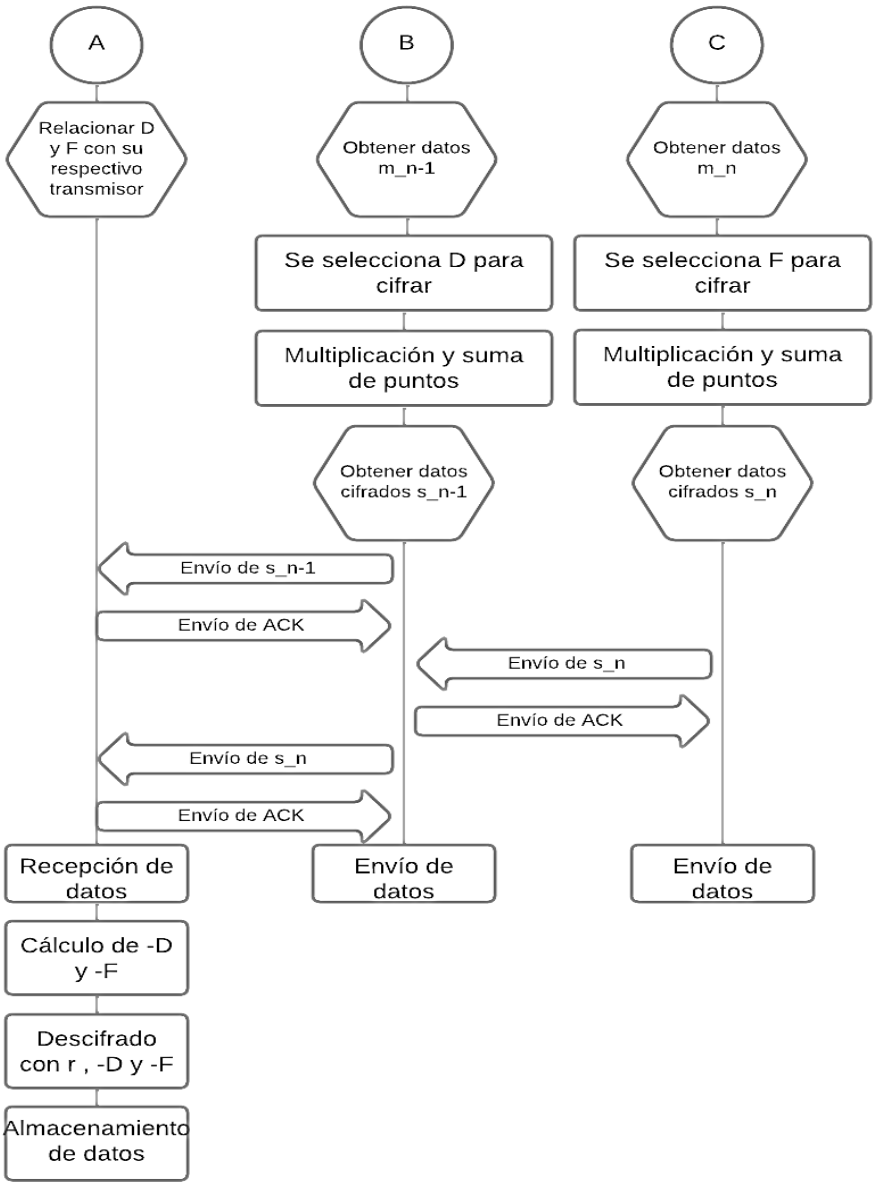


Figura 15
Diagrama de Secuencia del cifrado



CAPÍTULO IV

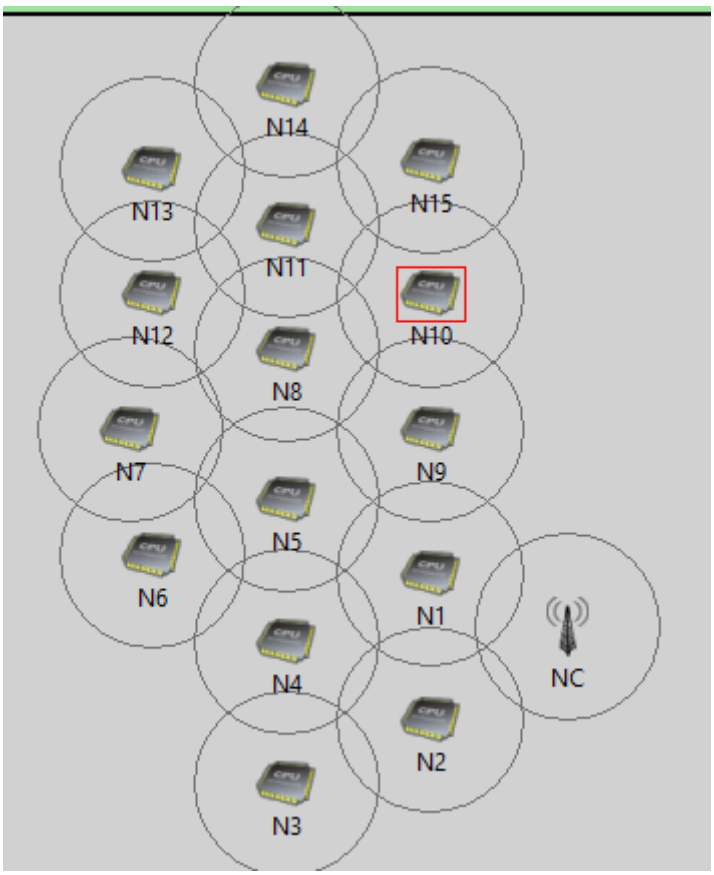
Implementación y pruebas

En este capítulo se podrá evidenciar el proceso de cómo se llega a la solución, cómo se configuró la red con la respectiva implementación y adhesión del algoritmo. Además, se realizan las pruebas necesarias para demostrar la funcionalidad de la solución.

4.1 Codificación

Para el desarrollo de la simulación de una Red de Sensores en el software OMNet++ se toma como referencia la simulación de la red WSN diseñada para el Bosque Protector Guayabillas (Enríquez Burgos, 2018) en la cual están establecidos los parámetros de la red y su funcionamiento, lo que marca el punto de inicio para la simulación y posteriormente el cifrado.

Figura 16
Captura del archivo .ned



4.1.1 Configuración de la Red

Al iniciar la programación se crea un archivo con extensión .ned en la cual se puede diseñar la red configurando los nodos de manera gráfica o en código fuente. En la Figura 16 se muestra una captura de pantalla del despliegue de la red en el archivo .ned en forma gráfica, y en la Figura 17 se muestra de manera general el código fuente de la configuración de características del canal de comunicación usado en el que se determina el valor de velocidad de transmisión del protocolo base que es XBee, esto con la finalidad de que el algoritmo se encargue de buscar y utilizar el canal más adecuado para mejorar el rendimiento de la red.

Figura 17

Configuración del canal de comunicación

```
channel C extends ned.DatarateChannel
{
    parameters:

        volatile double jitter @unit(ms) = default(uniform(0ms,6 ms));
        delay = 2ms + convertUnit(jitter, "ms");
        datarate = default(0.25Mbps);   //velocidad de transmisión según el protocolo XBee 250.000 Kbps
        @display("ls= grey,0");
        double noise @unit("dB") = default(1dB);
}
```

En la Figura 18 se muestra el proceso de creación y configuración de cada nodo, en total se configuraron 15 nodos y un nodo Central denominado ‘NC’.

Figura 18

Configuración de características de cada nodo

```
submodules:

    NC: Terminal {
        @display("p=546,396;r=60,,#707070;i=device/antennatower");
        address = 1;
        gates:
            port[8];
    }

    N1: Terminal {
        address = 16;
        @display("p=457,361;r=60,,#707070;i=device/cpu");
        gates:
            port[8];
    }

    N2: Terminal {
        @display("p=457,456;r=60,,#707070;i=device/cpu");
        address = 2;

        gates:
            port[8];
    }

    N3: Terminal {
        @display("p=364,498;r=60,,#707070;i=device/cpu");
        address = 3;
```

En la Figura 19 se puede observar cómo se configuran las conexiones entre nodos y se finaliza con el ciclo que realiza cada nodo para realizar sus funciones de envío, recepción y transmisión de datos.

Figura 19

Conexiones entre nodos

```
N8.port[4] <--> C <--> N7.port[1];
N8.port[5] <--> C <--> N12.port[0];
N11.port[3] <--> C <--> N14.port[1];
N11.port[4] <--> C <--> N12.port[1];
N11.port[5] <--> C <--> N13.port[0];
N14.port[2] <--> C <--> N13.port[1];
N6.port[2] <--> C <--> N7.port[2];
N7.port[3] <--> C <--> N12.port[2];
N12.port[3] <--> C <--> N13.port[2];

for i=0..height-1, for j=0..width-1 {
    node[i*width+j].port[0] <--> C <--> node[(i+1)*width+j].port[1] if i!=height-1;
    node[i*width+j].port[2] <--> C <--> node[(i*width+j)+1].port[3] if j!=width-1;
}
node[(height*width)-1].port[2] <--> C <--> N8.port[3] if (height*width)>0;
node[((height-2)*width)-1].port[2] <--> C <--> N1.port[3] if (height*width)>=3;
```

El archivo creado para la configuración preliminar de la red, generación y compartición de claves se denomina AppUDP con la extensión .cc en el cual se realiza la programación secuencial del procedimiento que realizan los nodos sensores. En la figura 20 se puede evidenciar la generación de datos para cada nodo, en este caso se enviará datos de humedad y temperatura.

Figura 20

Generación de datos en cada nodo

```
if (msg == dataGenerate) // Generar paquetes
    {
        bool setup = false;
        scheduleAt(simTime()+sendIATime->doubleValue(),dataGenerate);
        // Enviar el siguiente paquete
        double comp = simTime().dbl();
        int tem = rand() % 45; //Generar datos de temperatura (random)
        int hum = rand() %100; //Generar datos de humedad (random)
```

Para la encriptación se realiza el proceso de inicialización de variables para las claves pública y privada, tal como se muestra en la Figura 21. Además, para generar la clave privada se utiliza un valor aleatorio con el fin de que ésta se actualice periódicamente.

4.1.2 Generación de claves

Figura 21
Variables para la encriptación

```
int privateKey; //Crear variables para la clave privada y las claves públicas
int publicKeyX;
int publicKeyY;
privateKey = rand() %1000; // Generar la clave privada (random)
```

A continuación, se procede a definir los parámetros que serán utilizados en la curva elíptica, tales son a, b que son números definidos por la ecuación, y p que es un número primo con el cual se realizarán los cálculos. La Figura 22 muestra esta configuración.

Figura 22
Definición de parámetros de la curva elíptica

```
//--------------Parametros curva eliptica -------------------------------

int a = 2; // parametro a de la curva
int b = -11; // parametro b de la curva
int p = 3847; //tamaño del campo Zp
int K = 10; // Rango para el cifrado
vector<int> PuntoGen = {343, 3206};
```

Para la generación de claves se utiliza un ciclo que realiza las operaciones en cada nodo como se muestra en la Figura 23.

Figura 23
Cálculo y Generación de clave pública

```
// Cálculo de la llave pública (x, y) en la curva elíptica
int x = 0;
int y = 0;
for (int i = 1; i < p; i++)
{
    if ((i * i * i + a * i + b) % p == 0)
    {
        x = i;
        y = static_cast<int>(std::sqrt(i * i * i + a * i + b)) % p;
        break;
    }
}
publicKeyX = x;
publicKeyY = y;
int sharedSecretX = (publicKeyX * tem) % p;
int sharedSecretY = (publicKeyY * hum) % p;
```

Para compartir las claves se realiza el proceso mostrado en la Figura 24, en el cual se toma el paquete generado por el nodo y se establece la dirección a la que se encuentra destinado, que es el Gateway para que éste conozca la clave pública de cada uno de los nodos.

Figura 24
Compartición de clave pública

```
//Compartición de claves

if (comp < (setupTime->doubleValue()) ) {
    setup = true; // skip send data packets
    return;
} else {
    int destAddress = destAddresses[intuniform(0, destAddresses.size()-1)];
    char pkname[40];
    //sprintf(pkname,"pk-%d-to-%d-#%ld-temp-%d-hum-%d", myAddress, destAddress, pkCounter++,tem,hu);
    sprintf(pkname,"pk-%d-to-%d-sharedkeyX-%d-sharedkeyY-%d", myAddress, destAddress,sharedSecretX,sharedSecretY);

    Packet *pk = new Packet(pkname);
    pk->setByteLength(packetLengthBytes->longValue());
    pk->setSrcAddr(myAddress);
    pk->setDestAddr(destAddress);
    pk->setTransientNodesArraySize(maxArray->longValue()); // initialise to safe value
                                                // cuidar que los valores no salgan más altos que 100
    EV << "generating packet " << pkname << endl;
    numSent++;numData++;
    emit(sentPkSignal,pk);
    send(pk,"out");
}
setup = setup;
```

4.1.3 Operaciones de curva elíptica

En el archivo AppUdp se realiza la programación de los métodos a utilizarse en el proceso de cifrado y descifrado de los datos generados por cada nodo. Para las operaciones que se realizan en criptografía se inicia buscando un número entero que cumpla con la condición de ser un número primo, por lo tanto, se configura un método para realizar la comprobación de que el número seleccionado efectivamente sea un número primo. En la Figura 25 se visualiza el código de esta comprobación.

Figura 25
Método para comprobar si un número es primo

```
bool isPrime(int number){

    if(number < 2) return false;
    if(number == 2) return true;
    if(number % 2 == 0) return false;
    for(int i=3; (i*i)<=number; i+=2){
        if(number % i == 0 ) return false;
    }
    return true;
```

Otra operación importante es la suma de puntos de la curva elíptica, para esto se ha creado un método denominado sumaPuntos en el que se hace uso de parámetros como P, Q, a, b, p. Para su adecuado funcionamiento se tomó como referencia la suma de puntos configurada en (Ortega Christopher, 2023). La operación de suma de puntos en una curva elíptica para criptografía se muestra en la Figura 26.

Figura 26

Operación Suma de Puntos

```
vector<int> sumaPuntos(vector<int> P, vector<int> Q, int a, int b, int p) {

    vector<int> R;
    int m = 0;

    if (a%1 != 0 || b%1 != 0) {
        cout << "El valor de a y de b deben ser enteros" << endl;
    } else if (isPrime(p) != 1 || p < 4) {
        cout << "p debe ser un numero primo mayor que 3" << endl;
    } else if ((4*pow(a,3) + 27*pow(b,2)) == 0) {
        cout << "Curva singular, Intenta otra curva" << endl;
    }


    if (P[0] == 999999) {
        if (fmod(pow(Q[1],2),p) != fmod(((pow(Q[0],3)) + a*Q[0] + b),p)) {
            cout << "El punto Q debe pertenecer a la curva eliptica" << endl;
        } else {
            R = Q;
            return R;
        }
    }


    if (Q[0] == 999999) {
        if (fmod((pow(P[1],2)),p) != fmod(((pow(P[0],3)) + a*P[0] + b),p)){
            cout << "El punto P debe pertenecer a la curva eliptica" << endl;
        } else {
            R = P;
```

Otro método utilizado en el algoritmo es la conversión de los datos en puntos de la curva elíptica, con esto se da guía para realizar las operaciones de cifrado y descifrado siguiendo las instrucciones y restricciones de la ECC. En la Figura 27 se muestra el método denominado charToPuntoEcc.

Figura 27

Método para convertir datos en puntos de la EC

```
vector<int> charToPuntoEcc(char m, int a, int b, int p, int K) {
    vector<int> M;
    if (a%1 != 0 || b%1 != 0) {
        cout << "El valor de a y de b deben ser enteros" << endl;
    } else if (isPrime(p) != 1 || p < 4) {
        cout << "p debe ser un numero primo mayor que 3" << endl;
    } else if ((4*pow(a,3) + 27*pow(b,2)) == 0) {
        cout << "Curva singular, Intenta otra curva" << endl;
    }

    double m_d = double(m);
    cout << m_d << endl;

    if ((m_d + 1)*K >= p) {
        cout << "Error: Con los valores de K y m, debe usar un primo p mas grande o un K menor" << endl;
        vector<int> error { 0, 0 };
        return error;
```

Para llevar a cabo el descifrado se requiere el proceso opuesto al método anterior, es decir, la conversión de puntos de la EC a datos, para esto se muestra en la Figura 28 el método puntoEccToChar.

Figura 28

Método para convertir un punto de la EC en datos.

```
char puntoEccToChar(vector<int> P, int a, int b, int p, int K) {
    int m = 0;
    char m_char;

    if (a%1 != 0 || b%1 != 0) {
        cout << "El valor de a y de b deben ser enteros" << endl;
    } else if (isPrime(p) != 1 || p < 4) {
        cout << "p debe ser un numero primo mayor que 3" << endl;
    } else if ((4*pow(a,3) + 27*pow(b,2)) == 0) {
        cout << "Curva singular, Intenta otra curva" << endl;
    } else if (int(pow(P[1],2))%p != (int(pow(P[0],3)) + a*P[0] + b)%p) {
        cout << "P debe pertenecer a la curva eliptica" << endl;
    }

    for (int i = 0; i < (K - 1); i++) {
        m = (((P[0] - i)/K)%p + p)%p;
        int x = ((m*K + i)%p + p)%p;
        if (P[0] == x) {
            m_char = char(m);
            return m_char;
```

Para el cifrado se genera un método denominado cifradorEcc2 en el cual se realizan las operaciones de encriptación utilizando los métodos de operaciones en curvas elípticas descritos anteriormente, la Figura 29 muestra el proceso.

Figura 29

Proceso de cifrado

```
vector<vector<int> > cifradorEcc2(int a, int b, int p, int K, vector<int> S, string msj) {
    vector<vector<int> > M;
    vector<vector<int> > C2;
    const int msj_length = msj.length();
    char* char_array = new char[msj_length + 1];
    strcpy(char_array, msj.c_str());
    //cout << "M: " << endl;
    for (int i = 0; i < msj_length; i++) {
        M.push_back(charToPuntoEcc(char_array[i], a, b, p, K));
        //cout << M[i][0] << " " << M[i][1] << endl;
    }

    int tam = M.size();

    for (int j = 0; j < tam; j++) {
        vector<int> M_temp { M[j][0], M[j][1] };
        C2.push_back(sumaPuntos(M_temp, S, a, b, p));
    }

    return C2; //MsjCifrado
`
```

En cuanto al descifrado se genera un método denominado descifradorEcc2 en el cual se realizan las operaciones opuestas a las de encriptación utilizando los métodos de operaciones en curvas elípticas ya descritos, la Figura 30 muestra el proceso.

Figura 30

Proceso de descifrado

```
string descifradorEcc2(int a, int b, int p, int K, vector<int> S, vector<vector<int> > MsjCifrado) {
    vector<vector<int> > C2 = MsjCifrado;
    vector<vector<int> > M;
    vector<int> mS { S[0], ((-S[1]%p)+ p)%p};
    int tam = C2.size();
    for (int i = 0; i < tam; i++) {
        vector<int> C2_temp { C2[i][0], C2[i][1] };
        M.push_back(sumaPuntos(C2_temp, mS, a, b, p));
    }

    string msjDescifrado = "";
    tam = M.size();
    for (int j = 0; j < tam; j++) {
        vector<int> M_temp { M[j][0], M[j][1] };
        char c = puntoEccToChar(M_temp, a, b, p, K);
        string s(1, c);
        msjDescifrado = msjDescifrado + s;
    }

    return msjDescifrado;
```

4.1.4 Cifrado

Para cifrar los datos se hace uso de los parámetros de curva elíptica adecuados para aplicaciones similares a la que se está analizando como es Redes de Sensores en sistemas de Alerta Temprana, establecidos en (Daniel R. L. Brown, 2017) los cuales se han configurado como se muestra en la Figura 31.

Figura 31
Parámetros de la curva elíptica

```
int a = 2; // parametro a de la curva
int b = -11; // parametro b de la curva
int p = 3847; //tamaño del campo Zp
int K = 10; // Rango para el cifrado
vector<int> PuntoGen = {343, 3206};
```

El proceso de cifrado realiza llamadas a los métodos de operaciones en curvas elípticas ya mencionados y se ejecuta como se muestra en la Figura 32.

Figura 32
Proceso de cifrado en los nodos

```
vector<vector<int> > cifradotemp = cifradorEcc2(a, b, p, K, PuntoGen, msg1_str);
vector<vector<int> > cifradohum = cifradorEcc2(a, b, p, K, PuntoGen, msg2_str);

EV << "----------------------------------------"<<endl;
EV << "**********CIFRANDO CON ECC**********" << endl;
EV << "----------------------------------------"<<endl;
EV << "CIFRADO PARA LA TEMPERATURA: " << endl;
for (unsigned int cont = 0; cont < cifradotemp.size(); cont++){
    EV << cifradotemp[cont][0] << " " << cifradotemp[cont][1] << endl;
}
string temperatura = puntostotext(a, b, p, K, cifradotemp);
EV << "Texto:" << temperatura << endl;
EV << "CIFRADO PARA LA HUMEDAD: " << endl;
for (unsigned int cont = 0; cont < cifradohum.size(); cont++){
    EV << cifradohum[cont][0] << " " << cifradohum[cont][1] << endl;
}
string humedad = puntostotext(a, b, p, K, cifradohum);
EV << "Texto:" << humedad << endl;
EV <<"----------------------------------------"<<endl;
if (comp < (setupTime->doubleValue()) ) {
    setup = true; // skip send data packets
    return;
} else {
```

4.1.5 Descifrado

Para la configuración de recepción de datos en el nodo central se crea un archivo de extensión .cc denominado UDPtrans en el cual se realiza el proceso de descifrado de la información generada por los nodos y recibida por el nodo central. En la Figura 33 se muestra el código con el cual el nodo central realiza la lectura del paquete que llega y lo convierte en una cadena string.

Figura 33

Lectura del paquete recibido por el nodo central

```
void UDPtrans::handleMessage(cMessage *msg)
{
    vector<vector<int>> tempVector;
    vector<vector<int>> moistVector;

    // received from IP layer
    if (msg->arrivedOn("ipIn")) // || msg->arrivedOn("ipv6In"))
    {
            processUDPPacket(msg); // pasar de transformar en paquete UDP
            const char* messageContent = msg->getFullName();
            std::string myString(messageContent);
            std::vector<std::string> values;
            std::istringstream ss(myString);
            std::string token;   // Caracter separador de datos '-'
            std::string temp;    //Dato de temperatura
            std::string moist;   //Dato de humedad
            int source;
            std::string type;
```

A continuación, se hace uso de un vector para ubicar los datos de la variable string, también se genera un ‘token’, carácter que separa los datos y como resultado las variables son separadas y ubicadas adecuadamente en el vector como se muestra en el código de la Figura 34.

Figura 34

Separación de la variable tipo string a vector

```
const char* messageContent = msg->getFullName();
std::string myString(messageContent);

std::vector<std::string> values;
std::istringstream ss(myString);
std::string token;
int sharedX;
int sharedY;
int source;
std::string type;

while (std::getline(ss, token, '-')) {
    values.push_back(token);
}

if (values.size() >= 7) {
    std::string value1 = values[0];
    std::string value2 = values[1];
    std::string value3 = values[2];
    std::string value4 = values[3];
    std::string value5 = values[4];
    std::string value6 = values[5];
    std::string value7 = values[6];

    // Process the values in separate variables
    // ...
    sharedX = std::stoi(value5);
    sharedY = std::stoi(value7);
    source= std::stoi(value2);
    type=value1;
```

El nodo central NC realiza luego una conversión de los datos recibidos como variable string a datos de tipo decimal para ser interpretados luego como datos en texto plano como muestra la Figura 35.

Figura 35
Conversión de string a decimal

```
istringstream tempStream(temp);
int token;
while (tempStream >> token) {
    vector<int> pair;
    pair.push_back(token);
    tempStream >> token;
    pair.push_back(token);
    tempVector.push_back(pair);
}
istringstream moistStream(moist);
int token2;
while (moistStream >> token2) {
    vector<int> pair;
    pair.push_back(token2);
    moistStream >> token2;
    pair.push_back(token2);
    moistVector.push_back(pair);
}
```

Figura 36
Configuración de salida en pantalla de los datos descifrados

```
if (type=="pk")
{
 //---------------------Recepcion de los paquetes----------------------------
 std::string result = "Paquete del Nodo #"+ std::to_string(source) + " Recibido";
 getParentModule()->bubble(result.c_str());
 EV <<"---------------------------------------- "<< endl;
 EV << "----------- DATOS DEL NODO: "<< source <<" ----------" << endl;
 EV << "----------------------------------------"<<endl;
 EV << "Datos Recibidos: "<< temp << moist <<endl;
 EV << "----------------------------------------" << endl;
 EV << "**********DESCIFRANDO CON ECC***********" << endl;
 EV << "----------------------------------------" << endl;
 int a = 2; // parametro a de la curva
 int b = -11; // parametro b de la curva
 int p = 3847; //tamaño del campo Zp
 int K = 10; // Rango para el cifrado
 vector<int> PuntoGen = {343, 3206};

 string temp2 = descifradorEcc2(a, b, p, K, PuntoGen, tempVector);
 string moist2 = descifradorEcc2(a, b, p, K, PuntoGen, moistVector);

 EV <<"El Nodo # "<< source << endl;
 EV <<"Temperatura: "<< temp2 << " Grados" << endl;
 EV <<"Humedad: "<< moist2 << "%" << endl;
 EV <<"---------------------------------------- "<< endl;
```

En la Figura 36, se puede observar cómo el nodo central llama al proceso denominado descifradorecc2 que contiene los valores de temperatura y humedad con la finalidad de mostrar en pantalla estos datos ya descifrados y así se puedan comparar con los enviados de cada nodo

para comprobar que, en efecto, son los mismos. Así, se demuestra que el nodo central ha realizado adecuadamente el proceso opuesto al cifrado para obtener los datos que se generaron en los nodos. Para la demostración en la simulación, se configuró una impresión en texto, de manera que se pueda distinguir el proceso que se está realizando y sobre todo que permita la visualización clara de los datos desde cómo se generan en cada nodo y cómo llegan y se descifran en el nodo central.

4.2 Pruebas de Funcionamiento

En este apartado se procede a realizar las pruebas de funcionamiento de la solución en base a dos fases. En la primera Fase se realizarán Pruebas Preliminares, en las que se pondrá en evidencia tanto el funcionamiento de la red simulada como la generación y compartición de claves. En la Segunda Fase se realiza las pruebas de ejecución que engloba el cifrado y descifrado y además el consumo y carga que tiene el algoritmo con relación a la red.

4.2.1 Pruebas Preliminares

En esta sección se realizan dos tipos de prueba que demuestran la convergencia de la red y la generación/compartición de claves:

- Funcionamiento de la Red.
- Generación y compartición de claves.

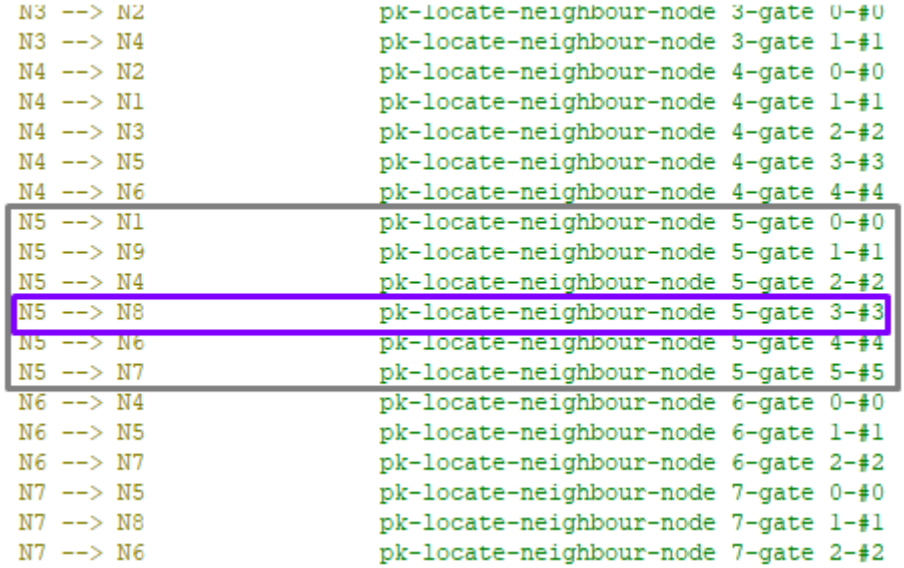
4.2.1.1 Funcionamiento de la Red

El funcionamiento de la red inicia con el reconocimiento de nodos vecinos por parte de cada nodo. Para lo cual se ha configurado un tipo de paquete denominado *pk-locate-neighbour-node* mediante el cual todos los nodos solicitan mensajes de confirmación a sus respectivos vecinos con el fin de reconocerse entre ellos. Este proceso lo realizan todos los nodos para cada

compuerta que se encuentre habilitada y conectada a un nodo vecino. Las compuertas (gate) son los puertos de conectividad por medio de los cuales se interconectan los nodos entre sí. Están configuradas de manera que cada nodo pueda crear un enlace hacia otro y así formar la red tipo malla tal como está diseñada en la investigación de referencia (Enríquez & Michilena, 2018). En la Figura 37 se muestra un fragmento de este proceso tomando en cuenta el reconocimiento de nodos vecinos que realiza cada nodo, por ejemplo, se destaca la tabla de vecinos del nodo 5 con color gris, en la primera columna el nodo identifica a cada uno de sus vecinos, en la segunda columna indica el nombre del nodo (node 5) y la compuerta (gate 3) por la que se comunica con el nodo vecino (N8) destacado en color azul.

Figura 37

Reconocimiento de Vecinos



En la Tabla12 se resume la cantidad de vecinos y se hace mención a todos los nodos que constan como vecinos para cada nodo. Como se puede observar existe interconexión entre todos los nodos y el Nodo Central al cual llegan todos mediante el Nodo 1 y el Nodo 2. Como se puede observar para cada nodo definido como vecino, existe una compuerta de conexión a ese nodo por lo que la cantidad de ‘gates’ debe ser la misma cantidad de nodos vecinos registrados.

Tabla 12
Tabla de vecinos para cada nodo

NODOS	***N° de Compuertas***	***Nodos Vecinos***
NC	2	N1, N2
N1	5	NC, N2, N9, N4, N5
N2	4	NC, N3, N4, N1
N3	2	N2, N4
N4	5	N2, N1, N3, N5, N6
N5	6	N1, N9, N4, N8, N6, N7
N6	3	N4, N5, N7
N7	4	N5, N8, N6, N12
N8	6	N9, N10, N5, N11, N7, N12
N9	4	N1, N10, N5, N8
N10	4	N9, N15, N8, N11
N11	6	N10, N15, N8, N14, N12, N13
N12	4	N8, N11, N7, N13
N13	3	N11, N14, N12
N14	3	N15, N11, N13
N15	3	N10, N11, N14

En cuanto al enrutamiento se hace uso de dos tipos de paquetes como FA que es el mensaje RREQ que envían los nodos a sus vecinos solicitando las rutas, y el BA (RREP) que son las respuestas a las solicitudes de rutas recibidas. La Figura 38 muestra esta comunicación entre algunos de los nodos, específicamente se destaca el caso del nodo 13.

Figura 38
Generación de rutas

Time	Src/Dest	Name
0.03	N7 --> N5	FA-7-to-1-#0
0.03	N13 --> N11	FA-13-to-1-#0
0.032266137581	N5 --> N1	FA-7-to-1-#0
0.033951836374	N11 --> N10	FA-13-to-1-#0
0.03549676796	N1 --> NC	FA-7-to-1-#0
0.038187846297	N10 --> N9	FA-13-to-1-#0
0.040127337111	NC --> N1	BA-1-to-7-#0
0.042905147722	N9 --> N1	FA-13-to-1-#0
0.045955068993	N1 --> N5	BA-1-to-7-#0
0.049123083057	N1 --> NC	FA-13-to-1-#0

La figura 39 muestra cómo el nodo 13 realiza los saltos necesarios hasta llegar al nodo central siguiendo la ruta marcada. La trama lleva identificadores de manera que cada nodo

pueda distinguir si es el destino del paquete o si debe pasarlo al siguiente salto y de esta manera completar el recorrido adecuadamente.

Figura 39
Ruta de un paquete enviado desde N13 hasta NC

Time	Src/Dest	Name
0.03	N13 --> N11	FA-13-to-1-#0
0.033951836374	N11 --> N10	FA-13-to-1-#0
0.038187846297	N10 --> N9	FA-13-to-1-#0
0.042905147722	N9 --> N1	FA-13-to-1-#0
0.049123083057	N1 --> NC	FA-13-to-1-#0

En la simulación se evidencia que todos los nodos tienen comunicación entre sí y con el nodo central, realizando el envío de datos aun sin encriptar a manera de texto plano como se muestra en la Figura 40, de esta manera el Nodo 1 envía un valor de '13' para Temperatura y un valor de '3' para Humedad. En la Figura 41 se muestra cómo están llegando los datos al Nodo Central de igual manera en texto plano y se puede observar que los datos de Temperatura y Humedad son exactamente los mismo que generó y envió el nodo 1. Según las rutas establecidas los datos viajan desde el origen al destino valiéndose del enrutamiento configurado por el cual los nodos que no se identifican como destinatarios realizan una función de Forwarding con cada paquete.

Figura 40
Envío de datos en texto plano desde el nodo 1

```
----------------------------------------
----------------NODO: 1---------------
----------------------------------------
Obteniendo datos de los sensores
----------------------------------------
Temperatura 13
Humedad 3
----------------------------------------
```

Figura 41
Recepción de datos en texto plano en el Nodo Central.

```
----------------------------------------
El Nodo # 1
Temperatura: 13
Humedad: 3
----------------------------------------
```

4.2.1.2 Generación y compartición de claves

En el caso de generación y compartición de claves, en el software OMNet++ se ejecuta una serie de instrucciones que permite evidenciar que un sensor ha calculado y generado sus claves. A manera de ejemplo, en el nodo 13 se genera una clave pública que es compartida mediante una ruta establecida por saltos entre los sensores hasta llegar al nodo central como se muestra en la Figura 42.

Figura 42

Ruta de envío de la clave pública desde el nodo 13 al nodo central

```
N13 --> N11          pk-13-1-sharedkeyX-322-sharedkeyY-24472
N11 --> N10          pk-13-1-sharedkeyX-322-sharedkeyY-24472
N10 --> N9           pk-13-1-sharedkeyX-322-sharedkeyY-24472
N9 --> N1            pk-13-1-sharedkeyX-322-sharedkeyY-24472
N1 --> NC            pk-13-1-sharedkeyX-322-sharedkeyY-24472
```

4.2.2 Pruebas de funcionamiento del algoritmo

El propósito de esta sección es evidenciar el funcionamiento del algoritmo, para lo cual se toma como referencia el nodo 13 para demostrar que el nodo realiza el cifrado y que el nodo central está realizando el descifrado.

4.2.2.1 Cifrado

Para el cifrado el Nodo 13 obtiene los datos de temperatura y humedad, luego los cifra y realiza el envío de estos en forma de puntos que se encuentran dentro de la curva elíptica establecida. Según la ruta establecida en el simulador, este paquete debe dar cinco saltos antes de llegar a destino. Cabe recalcar que la información al ser transportada va codificada por lo tanto solamente el nodo central puede descifrar los datos. El proceso de generación de datos, cifrado y envío se resume en la Figura 43.

Figura 43
Cifrado en el Nodo 13

```
----------------------------------------
----------------NODO: 13----------------
----------------------------------------
Obteniendo datos de los sensores
----------------------------------------
Temperatura: 36 Grados
Humedad 41%
----------------------------------------
----------------------------------------
**********CIFRANDO CON ECC*************
----------------------------------------
CIFRADO PARA LA TEMPERATURA:
1826 437
1268 3438
Texto:□~
CIFRADO PARA LA HUMEDAD:
1342 2773
1781 3583
Texto:□□
----------------------------------------
```

La trama que se envía desde el nodo origen (N13) con datos cifrados por el medio de transmisión queda estructurada como se observa en la Figura 44, todos los nodos reciben la trama, pero los datos se perciben en forma de puntos, que serán interpretados únicamente por el nodo central.

Figura 44
Trama de datos cifrados que se envía a través de la red.

```
----------------------
Generando paquete pk-13-to-0-#0-temp-1826 437 1268 3438 -hum-1342 2773 1781 3583
----------------------
```

4.2.2.2 Descifrado

El Nodo Central recibe la trama que atravesó toda la red con los datos cifrados, en la Figura 45 se puede observar que los datos llegan al nodo central diferenciados en cadenas diferentes para temperatura y humedad pero están cifrados en forma de puntos. Además el nodo central recibe la información de cuántos saltos ha dado el paquete para llegar al destino.

Figura 45
Trama recibida en el Nodo Central con origen en el Nodo 13

```
pk-13-to-0-#0-temp-1826 437 1268 3438 -hum-1342 2773 1781 3583  despues de 5 saltos
```

El nodo central ejecuta el proceso opuesto al que realizan los nodos para cifrar, en la figura 46 se muestra los datos que obtiene el Nodo Central luego de descifrar la trama.

Figura 46
Datos Descifrados en el Nodo Central provenientes del Nodo 13

```
----------------------------------------
---------- DATOS DEL NODO: 13 ----------
----------------------------------------
Datos Recibidos: 1826 437 1268 3438 1342 2773 1781 3583
----------------------------------------
**********DESCIFRANDO CON ECC***********
----------------------------------------
El Nodo # 13
Temperatura: 36 Grados
Humedad: 41%
----------------------------------------
```

Se puede evidenciar que los datos enviados por el Nodo 13 corresponden a los valores de 36 para temperatura y de 41 para humedad como se muestra en la Figura 43; estos datos al ejecutar el proceso de cifrado, envío y descifrado, son obtenidos en el Nodo Central como muestra la Figura 45 y luego de realizar el descifrado los datos obtenidos se muestran en la Figura 46. De esta manera se demuestra que tanto el proceso de cifrado como el descifrado están funcionando adecuadamente.

4.2.3 Registro de consumo de recursos

Para evidenciar el consumo del algoritmo se registra el tiempo de ejecución del proceso, el número de ciclos por reloj y además se muestra el número de instrucciones por segundo.

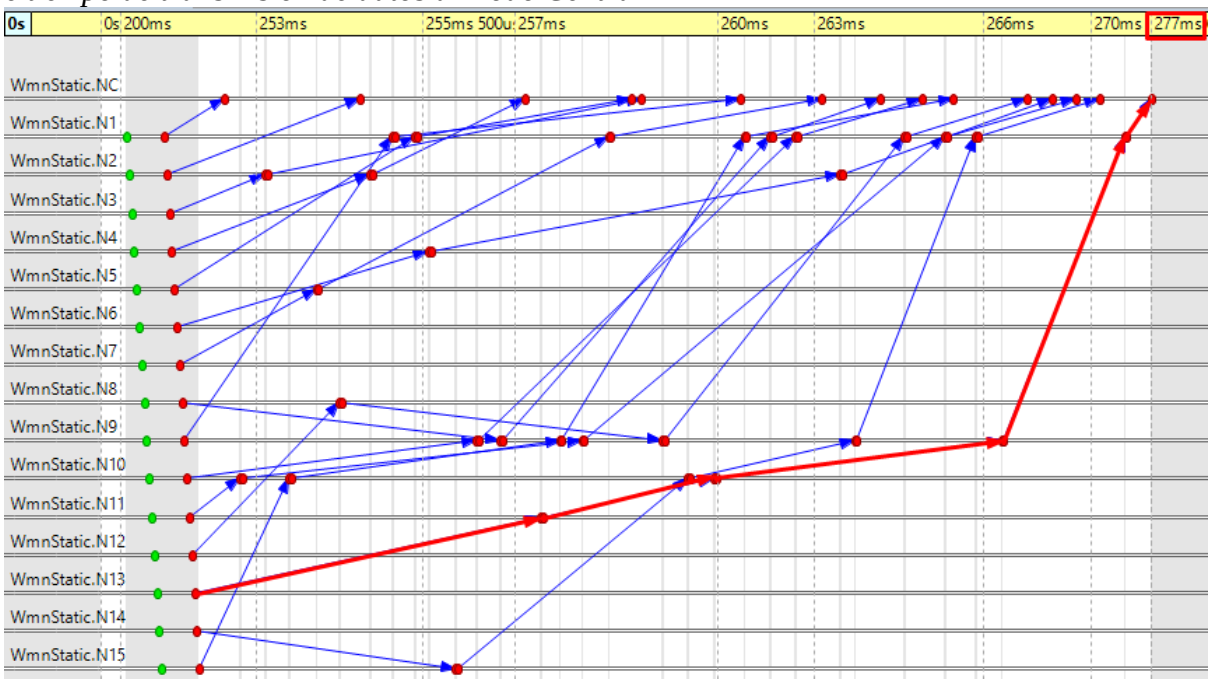
4.2.3.1 Tiempo de Ejecución sin cifrado

Para establecer el tiempo de ejecución del proceso se observa en la simulación el tiempo que se demoran todos los nodos en enviar sus respectivos datos sumado al tiempo en el que llegan al Nodo Central y éste los interpreta. En la Figura 47 se puede visualizar cómo los datos de cada nodo llegan al nodo central. El tiempo de llegada del paquete proveniente del nodo 13 es de 277 ms, éste es el nodo más lejano por lo que su paquete es el último en llegar al nodo

central, la ruta está marcada de color rojo. Además, se puede observar los saltos que deben dar los paquetes de cada nodo hasta llegar a su destino.

Figura 47

Línea de tiempo de transmisión de datos al Nodo Central



4.2.3.2 Tiempo de ejecución luego de cifrar/descifrar

Una vez realizado el proceso de cifrado y descifrado se vuelve a correr la simulación para comprobar la diferencia que genera en tiempo este proceso comparado con el resultado obtenido antes de cifrar. En la Figura 48 se muestra el tiempo total requerido para generar datos en los sensores, cifrarlos, enviarlos y descifrarlos en el Nodo Central. Para el Nodo 13 este proceso se llevó a cabo en 290 ms, se puede observar que la ruta del paquete del nodo 13 está marcada con color rojo. Por lo tanto, es posible concluir que en comparación con el tiempo que tomó el transporte de datos sin cifrar, el cifrado no tiene un aumento significativo en tiempo, de lo cual se puede deducir que el delay general de la red no se vería alterado significativamente.

Figura 48

Tiempo de cifrado y descifrado de datos en el nodo central

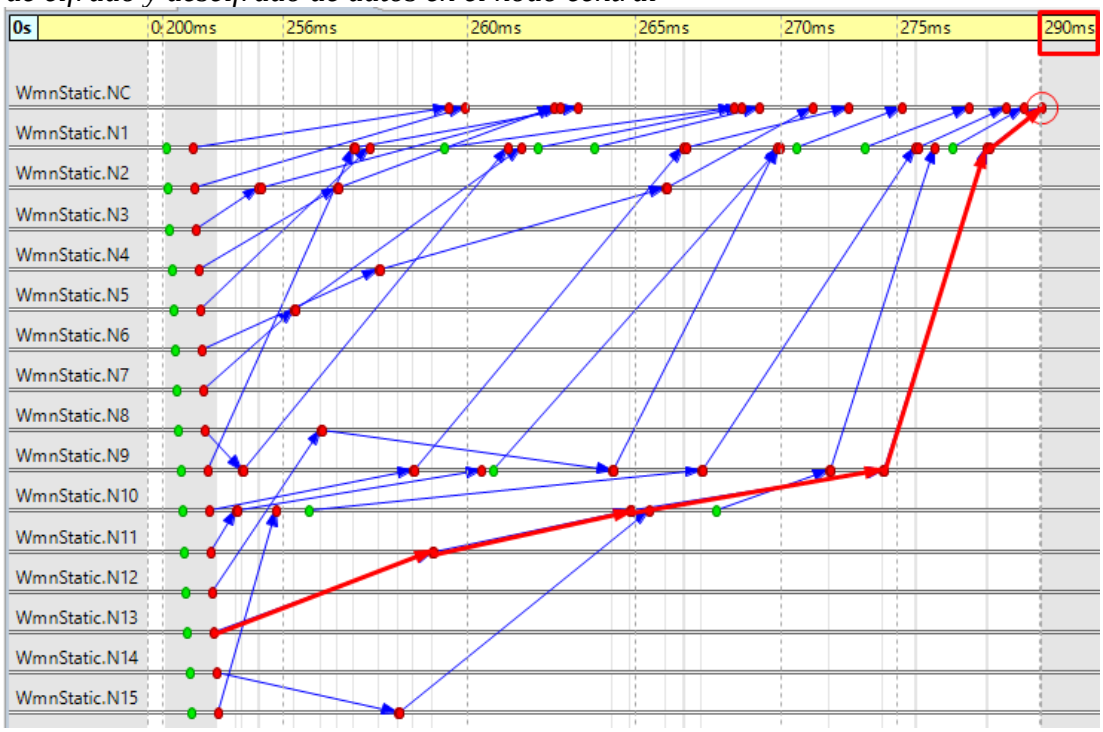


Tabla 13

Tiempo de transmisión de paquetes por nodo

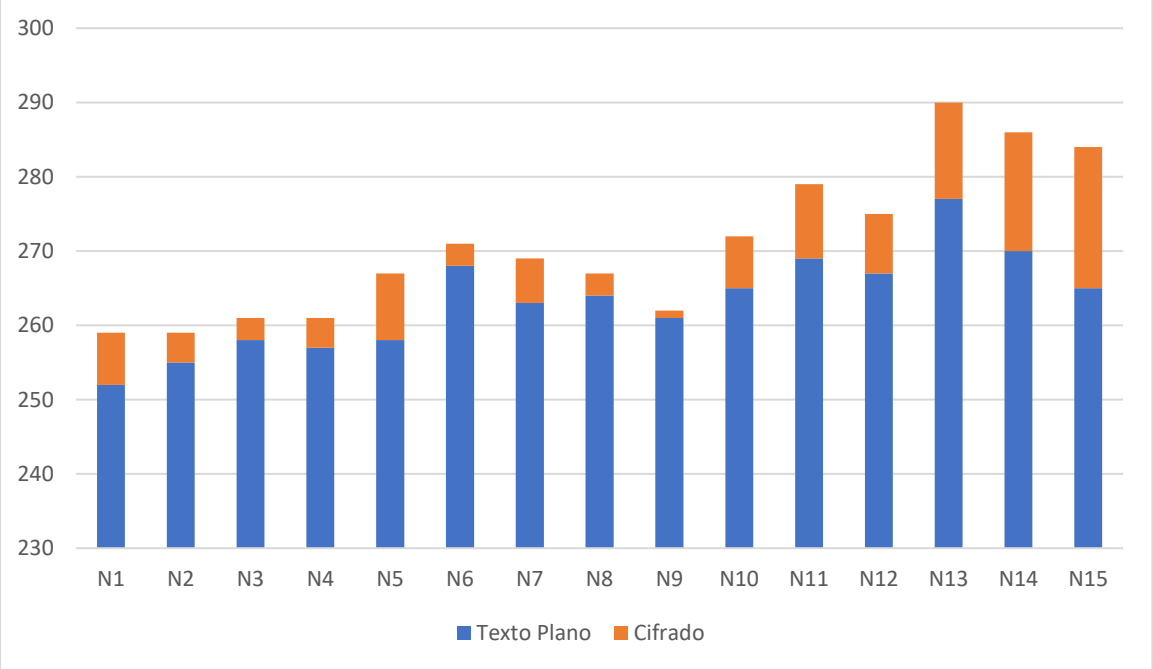
Nodos	Tiempo sin cifrado (ms)	Tiempo con cifrado (ms)	Tasa de retardo ECC
N1	252	259	2,78%
N2	255	259	1,57%
N3	258	261	1,16%
N4	257	261	1,56%
N5	258	267	3,49%
N6	268	271	1,12%
N7	263	269	2,28%
N8	264	267	1,14%
N9	261	262	0,38%
N10	265	272	2,64%
N11	269	279	3,72%
N12	267	275	3,00%
N13	277	290	4,69%
N14	270	286	5,93%
N15	265	284	7,17%

Adicionalmente, se ha elaborado una tabla comparativa con los dos valores de tiempo, tanto antes, como después de ejecutar el proceso de cifrado, y se ha calculado la tasa de afectación que el cifrado genera sobre cada nodo, esta información se muestra en la Tabla 13.

Es necesario realizar un análisis de los resultados obtenidos y para comprenderlos mejor se ha generado con ellos un gráfico de barras como muestra la Figura 49 en él se puede observar en qué medida aumenta el tiempo de transmisión para cada nodo al implementar el algoritmo criptográfico y enviar los datos cifrados. Las barras de color azul representan el tiempo de transmisión de datos sin cifrar, las barras de color naranja representan el aumento de tiempo, en milisegundos que genera el proceso de cifrado para cada nodo. Así se puede observar que el proceso de cifrado no representa alteraciones a la red, si bien, en los nodos más lejanos existe una diferencia mayor que en los nodos más cercanos, pero esto se debe al procesamiento que realizan los nodos intermedios al leer, identificar y enviar hacia el destino un paquete cifrado.

Figura 49

Incremento de tiempo de transmisión por nodo



4.2.3.3 Ciclos de reloj para el algoritmo

En este caso se calcula el número de ciclos de reloj que el algoritmo de ECC requiere para su funcionamiento, este dato puede ser útil al momento de querer implementar seguridad en una red física con el fin de seleccionar hardware y software que soporten este nivel de carga computacional. En este caso se realizan dos pruebas, el número de ciclos por reloj del proceso en un nodo que se muestra en la Figura 50, y, el resultado del mismo proceso, pero a nivel general en toda la red que se puede evidenciar en la Figura 51.

Figura 50
Ciclos de reloj en un nodo

```
Ciclos de reloj:
   515047680
```

Figura 51
Ciclos de reloj en toda la red

```
Ciclos de reloj:
   2.359le+08
```

4.2.3.4 Número de instrucciones por segundo del algoritmo

La cantidad de instrucciones por segundo que se llevan a cabo dependen de varios factores como la memoria, el procesador, la capacidad de almacenamiento y la velocidad de transmisión de los datos. La unidad de medida denominada MIPS (Millones de Instrucciones Por Segundo) indica exactamente la capacidad de procesar cierta cantidad de instrucciones en un segundo. De igual manera se obtiene la cantidad de instrucciones por segundo ejecutadas en un solo nodo como en la Figura 52 y también a nivel general de toda la red en lo que respecta a cifrado y descifrado en la Figura 53.

Figura 52
MIPS en un nodo

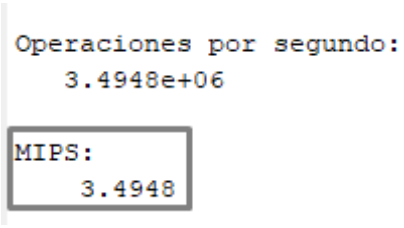
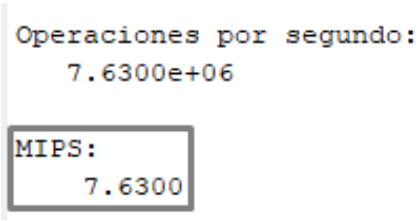


Figura 53
MIPS en toda la red



Finalmente se presenta un resumen de la implementación y la ejecución de pruebas descrito en la Tabla 14:

Tabla 14
Resumen de pruebas realizadas

Pruebas		*Logrado*	*Observación*
Pruebas Preliminares	Funcionamiento de la Red	Si	La red funciona adecuadamente con un canal de comunicación configurado a 250 000 Kbps. Los nodos generan datos randóm que los envían por la red y el nodo central los recibe distinguiendo adecuadamente el origen de cada paquete.
	Generación y compartición de claves	Si	Todos los nodos generan y envían sus claves al nodo central.
Pruebas de funcionamiento del algoritmo	Cifrado	Si	Los datos que se transmiten por la red no se pueden interpretar debido a que se está realizando adecuadamente el cifrado, la información pasa en forma de puntos de una curva elíptica desconocida para terceros.
	Descifrado	Si	El nodo central ejecuta las instrucciones opuestas a las operaciones de cifrado y logra descifrar los datos recibidos en forma de puntos, y convirtiéndolos en texto plano.
Registro de consumo de recursos	Tiempo de ejecución sin cifrado	Si	En el simulador OMNet++ se evidencia el tiempo que le toma a la red generar datos en todos los sensores, enviarlos y

			que el nodo central los reciba adecuadamente.
	Tiempo de Ejecución luego de cifrar/descifrar		Cuando se agrega el cifrado, el tiempo aumenta, pero no existe gran diferencia que pueda significar afectaciones al funcionamiento adecuado de la red como retardos, sobrecargas, o fallas en la transmisión.
	8Número de ciclos de reloj	Si	Esta variable se calcula tomando en cuenta también el valor del procesador que es de 1.8 GHz, el programa configurado permite realizar las pruebas con diferentes valores de procesador con el fin de conocer el comportamiento del algoritmo según los recursos disponibles.
	Instrucciones por segundo (MIPS)	Si	El programa calcula el número de instrucciones por segundo que según la cantidad de recursos disponibles se encuentra en el orden de MIPS

4.3 Discusión

La investigación desarrollada obtiene como resultados que en una red WSN puede implementarse un nivel de seguridad aceptable y de manera eficiente haciendo uso de algoritmos criptográficos de curvas elípticas, ya que los cálculos no generan gran consumo de recursos por lo que la red no se vería afectada en gran manera. Los resultados obtenidos coinciden con la investigación “UN ESQUEMA DE DISTRIBUCIÓN DE CLAVES QUE UTILIZA CRIPTOGRAFÍA DE CURVA ELÍPTICA EN REDES DE SENSORES INALÁMBRICOS” (Louw et al., 2016) en el uso de esquemas criptográficos de curvas elípticas para el intercambio de claves y para los procesos de cifrado y descifrado, utilizando ECDH y ECIES. Y concluyendo que si bien es cierto que la seguridad en WSN es un gran desafío, la ECC es una solución óptima que alcanza el mismo nivel de seguridad que otros algoritmos de clave pública como RSA sin necesidad de poner en riesgo el buen funcionamiento adecuado de la red y de los dispositivos. El presente proyecto de investigación también coincide con (Montoya, 2019) en que un esquema de ECC, en ambos casos ECDH,

proporcionan un buen nivel de seguridad y mejor escalabilidad en redes de recursos limitados basándose en el análisis de algunas limitantes propias de las redes IoT y que se asemejan a las limitantes de las WSN en sistemas de alerta temprana: consumo de procesamiento, consumo de energía, tiempo de ejecución, consumo de ancho de banda. También existe similitud entre los resultados de la presente investigación con los obtenidos en (Ravi et al., 2019), aunque la aplicación es distinta, los dos proyectos concluyen que para las WSN es recomendable utilizar algoritmos criptográficos de curvas elípticas principalmente por las restricciones y limitaciones de estas redes, esto se demuestra en que el delay de la red no registra alteraciones de gran medida al momento de ejecutar los procesos de cifrado y descifrado.

Conclusiones

- Al implementar el esquema de encriptación ECDH y el esquema de cifrado de curvas elípticas se logró configurar un algoritmo de seguridad de los datos con un nivel aceptable de robustez ya que los datos cifrados se transportan en la red en forma de puntos pertenecientes a una curva elíptica que solamente conocen el emisor y el receptor, esto además no implica un consumo excesivo de los recursos de la red lo que lo convierte en una solución óptima para las Redes de Sensores en este caso de Alerta Temprana.
- Luego de revisar toda la bibliografía disponible para comprender el funcionamiento de los algoritmos de Criptografía de Curvas Elípticas específicamente aplicados a Sistemas de Alerta Temprana en Redes WSN, no se encontró específicamente algo similar el caso de estudio del presente trabajo, entonces se optó por combinar las investigaciones existentes sobre seguridad en WSN y diversas aplicaciones de ECC para determinar las características del sistema que se implementó.
- Para determinar los requerimientos que permitieron la implementación del algoritmo de curva elíptica se elaboró una ficha en la cual se analiza cada requerimiento, de Stakeholders y del Sistema, describiendo cada uno y estableciendo su prioridad, el análisis se realizó en base a la investigación bibliográfica de casos de estudio relacionados con el tema siendo evaluados uno a uno por el autor y director del presente trabajo.
- En cuanto a la simulación se tomó como referencia una red WSN diseñada para el Bosque de Guayabillas de la ciudad de Ibarra, en la cual ya estaban definidas las características básicas del número de sensores, ubicación de nodos, métricas del ambiente y características de comunicación como tecnología y canal de transmisión, lo que sirvió de base para configurar los parámetros y lograr el funcionamiento de la red

en ambiente de simulación, de esta manera quedó apta para iniciar con la implementación de criptografía.

- La simulación se realizó en el software OMNet++ para demostrar cómo realiza la encriptación el algoritmo, en este caso hubo dificultades con las librerías de encriptación disponibles para ejecutar el cifrado, por lo tanto, las operaciones del algoritmo fueron programadas de forma manual.
- Para realizar la medición de la carga computacional se utilizó el software MatLab en el cual es posible evaluar parámetros de la red que no era posible evaluarlos en OMNet++, por lo que MatLab fue utilizado como software de apoyo para complementar la fase pruebas en lo que se refiere a carga computacional, se analizaron dos métricas: el número de ciclos por reloj y el número de instrucciones por segundo.
- Al realizar las pruebas se determinó que, en los parámetros evaluados en la implementación del algoritmo, la red no se ve afectada en gran manera por los procesos de cifrado y descifrado, lo que resulta en una ventaja para la ECC calificándola como apta para implementar seguridad en todo tipo de redes limitadas en software.

Recomendaciones

- Con los resultados obtenidos es posible determinar un valor aproximado de la capacidad de procesamiento requerida para implementar seguridad en una WSN, por lo que se sugiere, si es el caso de realizar pruebas o implementar de forma física que los nodos cuenten con la capacidad mínima de procesamiento para que el hardware y software no se excedan en procesamiento.

- Los resultados generados en la simulación relacionados con la carga computacional son de mucha ayuda ya que es necesario tomar en cuenta la capacidad de procesamiento de datos, la memoria y sobre todo la capacidad del procesador de los dispositivos, mediante la simulación es posible conocer el comportamiento del algoritmo si se varían las condiciones, por esta razón se recomienda ejecutar el proyecto en simulación antes de iniciar con una implementación física, ya conociendo los valores requeridos y según ellos seleccionar el hardware idóneo para una implementación.

- Si los datos de una red generan tramas de mayor tamaño, o se estima que un algoritmo criptográfico pone en riesgo el funcionamiento adecuado de la red, para este caso que se estudió una red WSN tipo malla, sería recomendable agregar en el otro extremo un nodo adicional que cumpla con las funciones de un segundo Nodo Central, de esta manera se evitaría que la red se congestione y los dos Nodos Centrales serían los que realicen el tratamiento de la información que reciban.

Referencias

Abo-Soliman, M. A., & Azer, M. A. (2018). A study in WPA2 enterprise recent attacks. *ICENCO 2017 - 13th International Computer Engineering Conference: Boundless Smart Societies*, *2018-Janua*, 323–330. https://doi.org/10.1109/ICENCO.2017.8289808

Ako, M. A. (2017). Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt Data. *Cryptography and Network Security*. https://www.researchgate.net/publication/317615794

Alblwi, S., Shujaee, K., & Atlanta, C. (2017). A Survey on wireless security protocol WPA2. *International Conference on Security and Management*, 12–17.

Aryan, Kumar, C., & Durai Raj Vincent, P. M. (2017). Enhanced diffie-hellman algorithm for reliable key exchange. *IOP Conference Series: Materials Science and Engineering*, *263*(4). https://doi.org/10.1088/1757-899X/263/4/042015

Aryeh, F. L., Asante, M., & Danso, A. E. Y. (2016). Securing Wireless Network Using pfSense Captive Portal with RADIUS Authentication – A Case Study at UMaT *. *Ghana Journal of Technology*, *1*(1), 40–45.

Báez, L. (2017). Revisión de las redes de sensores Inalámbricos (WSN) para el monitoreo automático de la calidad de agua subterránea. *Acuífero Patiño*, *01*.

Banđur, Đ., Jakšić, B., Banđur, M., & Jović, S. (2019). An analysis of energy efficiency in Wireless Sensor Networks (WSNs) applied in smart agriculture. *Computers and Electronics in Agriculture*, *156*(December 2018), 500–507. https://doi.org/10.1016/j.compag.2018.12.016

Barrera, E. G., & Whebe, R. A. (2020). Estudio comparativo de algoritmos de criptografía liviana. *Proyecto Final de Ingeniería*, *01*.

Bellalta, B. (2016). IEEE 802.11ax: High-efficiency WLANS. *IEEE Wireless Communications*, *23*(1), 38–46. https://doi.org/10.1109/MWC.2016.7422404

Cao, W., Shi, H., Chen, H., & Wang, Y. (2021). *A new weak curve fault attack on ECIES : embedded point validation is not enough during decryption*. 1–14.

Chen, M. Ed., Li, S., Wang, H., & Mobile, C. (2020). Use Identity as Raw Public Key in EAP - TLS draft-chen-emu-eap-tls-ibs-01. *Journal of Chemical Information and Modeling*, *53*(9), 1689–1699.

Chughtai, F., Ul Amin, R., Malik, A. S., & Saeed, N. (2019). Performance analysis of microsoft network policy server and freeRADIUS authentication systems in 802.1x based secured wired ethernet using PEAP. *International Arab Journal of Information Technology*, *16*(5), 862–870.

CISCO. (2017). *CISCO*. What Is Wi-Fi? https://www.cisco.com/c/en/us/products/wireless/what-is-wifi.html

Daniel R. L. Brown. (2017). *SEC 2: Recommended Elliptic Curve Domain Parameters*.

Dender-Zurita, R. C., Guilcatoma-Moreira, P. M., Argandoña-Moreira, J. G., & Machuca-Avalos, M. P. (2018). Gestión de redes de sensores inalámbricos (WSN) para la industria petrolera. *Polo Del Conocimiento*, *2*(12), 15. https://doi.org/10.23857/pc.v2i12.407

Enríquez, V., & Michilena, J. R. (2018). *Diseño de una red inalámbrica para una WSN de un sistema de alerta temprana de incendios para el Bosque Protector Guayabillas*.

Fernando, E., Agustin, D., Irsan, M., Murad, D. F., Rohayani, H., & Sujana, D. (2019). Performance Comparison of Symmetries Encryption Algorithm AES and des with Raspberry Pi. *Proceedings of 2019 4th International Conference on Sustainable Information Engineering and Technology, SIET 2019*, 353–357. https://doi.org/10.1109/SIET48054.2019.8986122

Granados, G. (2016). Introducción a la criptografía. *RDU - UNAM*, *7*(7). http://www.revista.unam.mx/vol.7/num7/art55/jul_art55.pdf

Hu, C., Han, L., & Yiu, S. M. (2016). *Efficient and secure multi-functional searchable*. *October 2015*, 34–42. https://doi.org/10.1002/sec

IETF, R. 1332. (1992). *RFC 1332*.

Jahan, S., Rahman, M. S., & Saha, S. (2017). Application specific tunneling protocol selection for Virtual Private Networks. *Proceedings of 2017 International Conference on Networking, Systems and Security, NSysS 2017*, 39–44. https://doi.org/10.1109/NSysS.2017.7885799

Karina, V. C., & Antonio, C. R. (2012). *Evaluación de implementaciones en software de algoritmos para la multiplicación escalar en criptografía de curvas elípticas*. *10*(1), 22–29.

Kaschel, H., & Abarzua, R. (2018). *Implementacion de una curva eliptica aplicada a una wsn limitada en hardware*. *November*.

Khanji, S., Iqbal, F., & Hung, P. (2019). ZigBee Security Vulnerabilities: Exploration and Evaluating. *2019 10th International Conference on Information and Communication Systems, ICICS 2019*, 52–57. https://doi.org/10.1109/IACS.2019.8809115

Lee, J., Su, Y., & Shen, C. (2017). A Comparative Study of Wireless Protocols : *IECON Proceedings (Industrial Electronics Conference)*, 46–51. https://doi.org/10.1109/IECON.2007.4460126

López Grande, C., & Guadrón Gutiérrez, R. (2015). Alta disponibilidad 24/7 : el reto. *REVISTA TECNOLÓGICA ITCA-FEPADE*, *2*.

Louw, J., Niezen, G., Ramotsoela, T. D., & Abu-Mahfouz, A. M. (2016). A key distribution scheme using elliptic curve cryptography in wireless sensor networks. *IEEE International Conference on Industrial Informatics (INDIN)*, *0*, 1166–1170. https://doi.org/10.1109/INDIN.2016.7819342

MathWorks. (2020). *Matlab. Matemáticas, Gráficas, Programación*. https://la.mathworks.com/products/matlab.html

Menezes, A., Vanstone, S., & Okamoto, T. (2011). Reducing elliptic curve logarithms to logarithms in a finite field. *Proceedings of the Annual ACM Symposium on Theory of Computing, Part F1300*, 80–89. https://doi.org/10.1145/103418.103434

Montoya, R. (2019). *Método Para La Preservación De La Privacidad En Dispositivos Iot Vesti-Bles Extendiendo La Seguridad Usando Fog Computing*. https://repositorio.itm.edu.co/handle/20.500.12622/1393

Mufida, E., Irawan, D., & Chrisnawati, G. (2017). Remote Site Mikrotik VPN Dengan Point To Point Tunneling Protocol (PPTP) Studi Kasus pada Yayasan Teratai Global Jakarta. *Jurnal Matrik*, *16*(2), 9. https://doi.org/10.30812/matrik.v16i2.7

Obaid, T., Rashed, H., -Elnour, A. A., Rehan, M., Muhammad Saleh, M., & Tarique, M. (2016). Zigbee Technology and its Application in Wireless Home Automation Systems: A Survey. *International Journal of Computer Networks & Communications*, *6*(4), 115–131. https://doi.org/10.5121/ijcnc.2014.6411

OpenSim Ltd. (2022). *https://omnetpp.org/intro/*. https://omnetpp.org/intro/

Ortega Christopher. (2023). *IMPLEMENTACIÓN DE MECANISMO DE SEGURIDAD PARA REDES DE SENSORES INALÁMBRICOS BASADO EN CRIPTOGRAFÍA DE CURVA ELÍPTICA*.

Parwinder, K. D. (2015). *Elliptic curve cryptography for real time embedded systems in IoT Networks*. *June*, 1–6.

Prakash, A., & Kumar, U. (2018). Authentication Protocols and Techniques A Survey. *International Journal of Computer Sciences and Engineering*, *6*(6), 1014–1020. https://doi.org/10.26438/ijcse/v6i6.10141020

Ravi, K., Khanai, R., & Praveen, K. (2019). Implementation of ECC on FPGA for WSN- a survey. *Proceedings of the 2019 IEEE International Conference on Communication and Signal Processing, ICCSP 2019*, 22–26. https://doi.org/10.1109/ICCSP.2019.8697952

Salazar, J. (2016). Redes Inalámbricas. In *Redes* (Vol. 2). http://www3.uah.es/vivatacademia/ficheros/n54/redesinalam.PDF

Sari, A., & Karay, M. (2015). Comparative Analysis of Wireless Security Protocols: WEP vs WPA. In *International Journal of Communications, Network and System Sciences* (Vol. 08, Issue 12, pp. 483–491). https://doi.org/10.4236/ijcns.2015.812043

Singh, K. K. V. V., & Gupta, H. (2016a). A new approach for the securityof VPN. *ACM International Conference Proceeding Series*, *04-05-Marc*(March 2016). https://doi.org/10.1145/2905055.2905219

Singh, K. K. V. V., & Gupta, H. (2016b). A new approach for the securityof VPN. *ACM International Conference Proceeding Series*, *04-05-Marc*(March 2016). https://doi.org/10.1145/2905055.2905219

University of Southern California. (2016). *The network Simulation ns2*. The Network Simulation Ns2. https://www.isi.edu/nsnam/ns/

Zaw, T. M., Thant, M., & Bezzateev, S. V. (2019). User Authentication in SSL Handshake Protocol with Zero-Knowledge Proof. *2018 Wave Electronics and Its Application in Information and Telecommunication Systems, WECONF 2018*, 1–8. https://doi.org/10.1109/WECONF.2018.8604392

Anexos

Anexo 1

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CIERCOM – CITEL

FICHA DE REQUERIMIENTOS

TEMA: "APLICACIÓN DE CRIPTOGRAFÍA DE CURVA ELÍPTICA PARA REDES DE SENSORES EN SISTEMAS DE ALERTA TEMPRANA"

TIEMPO DE ELABORACIÓN: 6 meses

FECHA: 02 de diciembre del 2022

LISTA DE STAKEHOLDERS

NOMENCLATURA DE REQUERIMIENTOS:

Requerimiento	Nomenclatura
De Stakeholders	*StSR*
De Sistema	*SySR*
De Arquitectura	*SrSH*

REQUERIMIENTOS DE STAKEHOLDERS:

Nomenclatura:	StSR 1
Requerimiento:	Estabilidad de la red simulada
Descripción:	Cuando se implemente la red en el ambiente de simulación es necesario que ésta sea estable, es decir que funcione adecuadamente el envío y recepción de paquetes de cualquier tipo sean mensajes "hello", o tramas de datos. Se debe destacar la importancia de este requerimiento antes de implementar el algoritmo de encriptación, de manera que al hacerlo no se tenga inconvenientes con la red.
Prioridad:	Alta

Nomenclatura:	StSR 2
Requerimiento:	Conocimiento en el lenguaje de programación
Descripción:	La simulación de redes de datos requiere cierto nivel de programación por lo tanto se necesita que el autor del presente trabajo investigativo esté capacitado para programar la red, asignarle características similares a la red base, monitorear la red y sobre todo introducir un algoritmo de encriptación en los nodos.
Prioridad:	Alta

Nomenclatura:	StSR3
Requerimiento:	**Conocimiento de Criptografía ECC**
Descripción:	Se requiere que para el adecuado desarrollo del proyecto los Stakeholders tengan conocimiento básico en el tipo de criptografía que se va a implementar de manera que, según el caso, se pueda ejecutar o guiar la investigación adecuadamente.
Prioridad:	Alta

Nomenclatura:	StSR 4
Requerimiento:	**Encriptación – Desencripción**
Descripción:	Son dos procesos importantes que al aplicarlos adecuadamente el autor puede comprobar si el algoritmo criptográfico logra transportar la información de forma segura desde cualquier nodo sensor hacia el nodo principal.
Prioridad:	Alta

Nomenclatura:	StSR 5
Requerimiento:	**Dispositivo base de alto rendimiento**
Descripción:	El investigador requiere un equipo sobre el cual instalar el simulador, este debe tener buenas características en cuanto a procesamiento y capacidad de computación para que la simulación con todos sus componentes pueda funcionar adecuadamente y lograr los objetivos propuestos para esta investigación.
Prioridad:	Media

Nomenclatura:	StSR 6
Requerimiento:	**Red WSN base**
Descripción:	Como punto de partida se requiere una red WSN que ya haya sido diseñada sobre la cual implementar el algoritmo de encriptación, ya que el presente trabajo tiene un enfoque de seguridad de la red.
Prioridad:	Alta

StSR				
No.	***Requerimiento***	***Prioridad***		
		Alta	***Media***	***Baja***
StSR 1	*Estabilidad de la red simulada*	*X*		
StSR 2	*Conocimiento en el lenguaje de programación*	*X*		
StSR 3	*Conocimiento en criptografía ECC*	*X*		
StSR 4	*Encriptación - Desencripción*	*X*		
StSR 5	*Dispositivo Base de alto rendimiento*		*X*	
StSR 6	*Red WSN base*	*X*		

REQUERIMIENTOS DEL SISTEMA:

Nomenclatura:	SySR 1
Requerimiento:	Red WSN en simulación
Descripción:	Es preciso configurar la red WSN en ambiente de simulación tomando como base las características de la red base y lograr que la adquisición y envío de datos del nodo sensor al nodo central se realice correctamente.
Prioridad:	Alta

Nomenclatura:	SySR 2
Requerimiento:	Medición de dato Temperatura
Descripción:	Los nodos sensores deben captar el valor de la variable Temperatura alojada en el sensor
Prioridad:	Media

Nomenclatura:	SySR 3
Requerimiento:	Medición de dato Humedad
Descripción:	Los nodos sensores deben captar el valor de la variable Humedad alojada en el sensor.
Prioridad:	Media

Nomenclatura:	SySR 4
Requerimiento:	Tratamiento de información
Descripción:	Los nodos sensores deben ser capaces de armar paquetes/tramas y enviar la información sensada hacia el nodo central
Prioridad:	Alta

Nomenclatura:	SySR 5
Requerimiento:	Monitoreo de parámetros
Descripción:	Cuando la red ya esté funcionando adecuadamente, se necesita aplicar técnicas de monitoreo de parámetros principales de la red con la finalidad de obtener información sobre el funcionamiento de la red antes y después de aplicar criptografía.
Prioridad:	Alta

Nomenclatura:	SySR 6
Requerimiento:	Simplicidad del Lenguaje de programación
Descripción:	El lenguaje de programación de preferencia debe ser amigable con el programador, de manera que sea comprensible lo que se ha programado.
Prioridad:	Alta

Nomenclatura:	SySR 7
Requerimiento:	Manuales de instalación y manejo
Descripción:	Para que el trabajo realizado pueda ser usado adecuadamente se puede elaborar manuales de usuario del manejo de la red con y sin criptografía.
Prioridad:	Media

Nomenclatura:	SySR 8
Requerimiento:	Menor tiempo de respuesta
Descripción:	Especialmente después de implementar el algoritmo, el tiempo de respuesta debe variar en cantidades mínimas para alcanzar los objetivos planteados. El retardo es un parámetro de suma importancia en redes de Alerta Temprana, el objetivo es que cuando se suscite un evento en el ambiente, estas redes envíen alarmas en el menor tiempo posible para que se pueda actuar a tiempo. Es decir, el tiempo de respuesta debe ser el mínimo posible. Si bien los cálculos para ejecutar el algoritmo deben ser complejos, difíciles de descifrar, también es muy importante que no exijan mucho de la capacidad de cómputo del nodo, de este modo el retardo que éste incremente al funcionamiento general del nodo será lo menor posible y no afecte en gran medida al retardo que ya posee la WSN.
Prioridad:	Media

Nomenclatura:	SySR 9
Requerimiento:	Seguridad
Descripción:	Además de comprobar si la criptografía no afecta en gran medida al rendimiento de la red, se debe encontrar un método que sirva para monitorear lo que sucede con las tramas y los datos.
Prioridad:	Alta

Nomenclatura:	SySR 10
Requerimiento:	Longitud de clave
Descripción:	En un algoritmo criptográfico normalmente la longitud de las claves generadas no se encuentra entre las prioridades, sin embargo, cuando se trata de redes WSN es un factor de alta prioridad puesto que según la longitud de la clave se consumirán memoria y capacidad de cómputo en cada nodo, por lo tanto al implementar un algoritmo criptográfico se debe procurar que utilice una longitud de clave mínima, con esto se pretende evitar que se eleve el consumo de memoria y de procesamiento.
Prioridad:	Alta

SySR				
No.	***Requerimiento***	***Prioridad***		
		Alta	*Media*	*Baja*
SySR 1	*Red WSN en simulación*	*X*		
SySR 2	*Medición de dato Temperatura*	*X*		
SySR 3	*Medición de dato Humedad*	*X*		
SySR 4	*Tratamiento de información*	*X*		
SySR 5	*Monitoreo de parámetros*	*X*		
SySR 6	*Simplicidad del Lenguaje de Programación*	*X*		
SySR 7	*Manuales de instalación y manejo*		*X*	
SySR 8	*Menor tiempo de respuesta*	*X*		
SySR 9	*Seguridad*	*X*		
SySR 10	*Longitud de clave*	*X*		

REVISADO POR	ELABORADO POR
MsC. Fabián Cuzme DIRECTOR DE TRABAJO DE GRADO	Andrea Ortega INVESTIGADOR