UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIA APLICADAS

CARRERA DE TELECOMUNICACIONES

TRABAJO DE INTEGRACIÓN CURRICULAR

TEMA:

"ADAPTACIÓN DE ALGORITMO CRIPTOGRÁFICO BASADO EN CURVAS ELÍPTICAS PARA REDES INALÁMBRICAS SUBACUÁTICAS"

Trabajo de titulación previo a la obtención del título en Ingeniero en Telecomunicaciones

Línea de investigación: Desarrollo, aplicación de software y cyber security (seguridad cibernética))

AUTOR:

Andrés Maurico Vásconez Folleco

DIRECTOR:

MSc. Fabián Geovánny Cuzme Rodríguez

Ibarra – Ecuador

2026

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:	1725612244		
APELLIDOS Y NOMBRES:	VÁSCONEZ FOLLECO ANDRÉS MAURICIO		
DIRECCIÓN:	Hugo Guzmán Lara n1-85 y José María Larrea y Jijón		
EMAIL:	amvasconezf@utn.edu.ec / mauriciovasco001@gmail.com		
TELÉFONO FIJO:	No registra	**TELÉFONO MÓVIL:**	0999934958

DATOS DE LA OBRA	
TÍTULO:	ADAPTACIÓN DE ALGORITMO CRIPTOGRÁFICO BASADO EN CURVAS ELÍPTICAS PARA REDES INALÁMBRICAS SUBACUÁTICAS.
AUTOR (ES):	VÁSCONEZ FOLLECO ANDRÉS MAURICIO
FECHA: DD/MM/AAAA	05/03/2026
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	■ **PREGRADO** □ **POSGRADO**
TITULO POR EL QUE OPTA:	INGENIERÍA EN TELECOMUNICACIONES
DIRECTOR:	MSC. FABIÁN GEOVANNY CUZME RODRÍGUEZ
ASESOR :	MSC. LUÍS EDILBERTO SUÁREZ ZAMBRANO

2. CONSTANCIAS

El autor (es) manifiesta (n) 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 (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 05 días del mes de marzo de 2026.

EL AUTOR:

Firma……

Firmado electrónicamente por:
ANDRES MAURICIO VASCONEZ FOLLECO
Validar únicamente con FirmaEC

……..

Nombre: Vásconez Folleco Andrés Mauricio

CERTIFICACIÓN DEL DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Ibarra, 05 de marzo del 2026

MSC. FABIÁN GEOVANNY CUZME RODRÍGUEZ

DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICA:

Haber revisado el presente informe final del trabajo de Integración Curricular, el mismo que se ajusta a las normas vigentes de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.

(f) …. 

……..

MSc. Fabián Geovanny Cuzme Rodríguez

DIRECTOR

C.C:1311527012

DEDICATORIA

Dedico este trabajo en primer lugar, a Dios. Mi primer agradecimiento es para él con el más profundo respeto y amor.

A mi madre Lourdes, pilares de mi vida, cuyo amor incondicional, esfuerzo silencioso, oraciones por mi bienestar y valores me han enseñado que la perseverancia y la honestidad son las bases de todo logro significativo. Sin su apoyo constante, nada de esto habría sido posible.

A mis docentes y mentores, por compartir generosamente sus conocimientos y por inspirarme a ir más allá de lo evidente, a cuestionar, crear y construir con propósito.

A mis amigos Fernando, Sergio, Luis y Anthony por su amistad incondicional, compañía, enseñanzas y aventuras durante mi experiencia universitaria.

Dedico también este logró respeto y gratitud, a todas aquellas personas que han sido parte fundamental de mi camino académico y personal como Jorge, Doris, Anita, Salomé y Abigail, Saira, Marcos, Consuelo, Lucia, Yolanda, Brad, Dianita y demás personas que me hay apoyado durante todo este proceso.

A mí mismo, Andrés Mauricio Vásconez Folleco, por no rendirme ante los desafíos, por confiar en mi capacidad de aprender, reinventarme y dejar una huella significativa a través de la tecnología y la investigación. Este logro representa no solo una meta académica, sino también la confirmación de que los sueños con propósito y disciplina se convierten en realidad.

Vasconez Folleco Andres Mauricio

AGRADECIMIENTO

Agradezco, en primer lugar, a Dios, fuente de fortaleza y guía constante en cada etapa de mi vida, por permitirme llegar hasta aquí con salud, determinación y claridad de propósito. Gracias a Dios he podido culminar un camino predestinado por él en cuanto a todas las circunstancias que han confirmado que este es su propósito en mi vida.

A mi madre porque realmente ha sido mi pilar, mi fortaleza y mi refugio ante todas las dificultades cotidianas, verdaderamente me siento tan bendecido de tenerla como mi madre. Y familia, por su amor incondicional, apoyo moral y sacrificio constante. Gracias por ser mi refugio en los momentos difíciles y por motivarme siempre a dar lo mejor de mí.

A la Universidad, por brindarme un espacio de formación académica y personal, así como por fomentar el pensamiento crítico y el compromiso con la investigación.

A mi director de tesis Msc Fabián y tribunal académico, por su orientación, tiempo y valiosos aportes durante el proceso investigativo. Sus observaciones fueron fundamentales para consolidar este proyecto.

A mis amigos incondicionales los cuales, no solamente hemos hecho trabajos juntos sino que hemos formado parte de un grupo muy bueno y lleno de mucha alegría en compartir cosas momentáneas y logros gratificantes.

Al Pastor Jorge Yandún y su familia porque me ayudaron de una manera tan incondicional, por lo cual deseo hacerles parte de este logro.

Este trabajo no solo representa la culminación de una etapa académica, sino también un reflejo del esfuerzo colectivo que me impulsó a alcanzar esta meta.

Vasconez Folleco Andres Mauricio

RESUMEN

La presente tesis adapta e integra un esquema criptográfico híbrido basado en criptografía de curvas elípticas para redes inalámbricas de sensores subacuáticos (UWSN), con el fin de asegurar confidencialidad, integridad y autenticación en un canal acústico de baja tasa, alta latencia y recursos limitados. Se implementó en C++ un módulo ligero que combina ECDH, KDF basada en SHA-256, cifrado AES-CTR con IV aleatorio por mensaje y firma ECDSA, integrado en OMNeT++ a nivel de aplicación bajo la política verify → decrypt.

La evaluación consideró seis escenarios (NS_Normal, NS_Listener, NS_Atacante y sus equivalentes seguros S_Normal, S_Listener, S_Atacante) con densidades N = 20, 50 y 100 y n = 3 repeticiones (54 corridas). Se midieron convergencia JOIN, retardo, PDR/PLR, throughput/goodput, overhead, descartes, tiempo de procesamiento y energía. Los resultados muestran el trade-off seguridad–eficiencia: sin seguridad se mantiene alta entrega (PDR ≈ 99%), mientras que con seguridad el overhead incrementa retardo y contención, reduciendo la entrega en redes densas (p. ej., S_Normal ≈ 96% en N=20 y ≈ 50% en N=100). El procesamiento permanece relativamente estable (≈ 0,58 s), por lo que la degradación se asocia principalmente a congestión del canal. Con atacante activo tipo flooding el desempeño colapsa por pérdida de disponibilidad, evidenciando la necesidad de mecanismos complementarios. Frente a listener pasivo, el rendimiento no cambia en NS pero el payload queda expuesto; en S permanece cifrado y autenticado.

Como limitaciones, el trabajo se basa en simulación, asume claves preconfiguradas y no incluye ataques criptográficos avanzados. Se propone como futuro trabajo incorporar distribución segura de claves, mecanismos MAC/controle de tráfico y validación mediante emulación o hardware.

Palabras clave: UWSN, canal acústico subacuático, criptografía de curvas elípticas, ECDH, ECDSA, AES-CTR, SHA-256, OMNeT++, seguridad, simulación.

ABSTRACT

This thesis adapts and integrates a hybrid elliptic-curve–based cryptographic scheme for underwater wireless sensor networks (UWSNs) to ensure confidentiality, integrity, and authentication over low-rate, high-latency acoustic channels. A lightweight C++ module combining ECDH, a SHA-256–based KDF, AES-CTR encryption with per-message random IV, and ECDSA signatures was integrated into OMNeT++ at the application layer under a verify → decrypt receiver policy.

The evaluation covered six scenarios (NS_Normal, NS_Listener, NS_Atacante and secure S_Normal, S_Listener, S_Atacante) across node densities N = 20, 50, and 100 with n = 3 repetitions (54 runs). Metrics included JOIN convergence, end-to-end delay, PDR/PLR, throughput/goodput, per-packet overhead, drops, processing time, and energy consumption. Results show a security–efficiency trade-off: without security, delivery remains high (PDR ≈ 99%), while security increases overhead and contention, reducing delivery at higher densities (e.g., S_Normal ≈ 96% at N=20 and ≈ 50% at N=100). Processing time stays relatively stable (≈ 0.58 s), suggesting that degradation is mainly driven by acoustic-channel congestion. Under an active flooding attacker, performance collapses due to loss of availability, indicating the need for complementary mitigation mechanisms.

As limitations, this study is simulation-based, assumes preconfigured keys, and does not model advanced cryptographic attacks. Future work includes secure key distribution, MAC/traffic-control mechanisms for availability, and validation through emulation or hardware experiments.

Keywords: UWSN, underwater acoustic channel, elliptic curve cryptography, ECDH, ECDSA, AES-CTR, SHA-256, OMNeT++, security, simulation.

INDICE DE CONTENIDOS

Contenido

INDICE DE FIGURAS

INDICE DE TABLAS

INDICE DE FORMULAS

Capítulo I: Antecedentes

1.1. Tema

ADAPTACIÓN DE ALGORITMO CRIPTOGRÁFICO BASADO EN CURVAS ELÍPTICAS PARA REDES INALÁMBRICAS SUBACUÁTICAS

1.2. Problema

La comunicación submarina presenta grandes desafíos debido a las características del entorno acuático como las altas tasas de error de bits, los retrasos de propagación variables y el bajo ancho de banda de los canales acústicos. Siendo más específicos la atenuación de la señal submarina puede reducir la calidad de la transmisión de datos y comprometer la seguridad de la comunicación al aumentar la posibilidad de interceptación por parte de terceros malintencionados (Kebapci et al., 2022). Además, las fluctuaciones en la transmisión, causadas por corrientes oceánicas u otros fenómenos, pueden afectar la integridad de los datos y dificultar la autenticación de los nodos en la red. Estas dificultades hacen que las redes de comunicaciones inalámbricas submarinas sean particularmente vulnerables a ataques maliciosos detalla (S.-H. Chang et al., 2015)

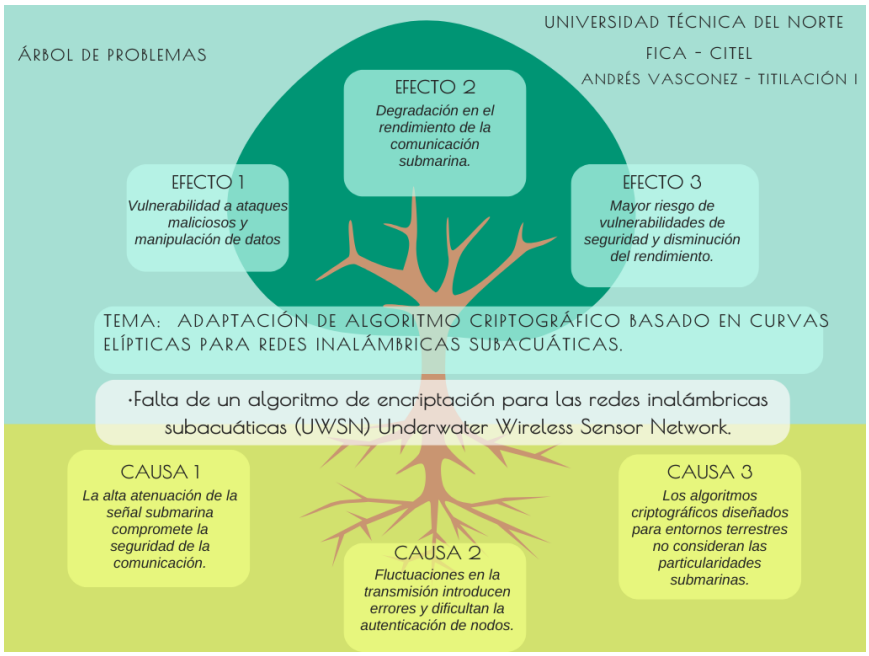
De la misma manera por la alta atenuación de la señal puede causar una degradación en el rendimiento de los sistemas de comunicación submarina enfocándose en velocidad y fiabilidad cuando hablamos de la seguridad de los datos que se están transmitiendo por este medio. Según (Cong et al., 2010) la incapacidad para mantener una conexión estable y de alta calidad puede comprometer la efectividad la confiabilidad e integridad de los datos y ponen en riesgo la seguridad de la red subacuática en su conjunto. Además, las fluctuaciones en la transmisión, causadas por factores como corrientes oceánicas y cambios en las condiciones ambientales, pueden introducir errores en los procesos de cifrado y descifrado, lo que aumenta el riesgo de vulnerabilidades de seguridad (A. A. Islam & Taher, 2021).

Los sistemas criptográficos desempeñan un papel fundamental en la seguridad de las redes inalámbricas subacuáticas al garantizar la confidencialidad, integridad y autenticidad de los datos transmitidos. Sin embargo, en el entorno subacuático, estos sistemas enfrentan desafíos únicos que limitan su eficacia y desempeño comenta (Haerbin, 2016). La alta atenuación de la señal, las fluctuaciones en la transmisión y las interferencias son algunos de los factores que pueden afectar la implementación y efectividad de los sistemas criptográficos en entornos subacuáticos.

También uno de los más grandes factores que estudia (A. A. Islam & Taher, 2021) es que la mayoría de los algoritmos criptográficos están diseñados para redes terrestres o aéreas, sin tener en cuenta las particularidades de la transmisión de datos submarinos debido a su reciente utilización. Esto puede llevar a vulnerabilidades de seguridad y a una disminución del rendimiento de la red cuando se implementan en redes inalámbricas subacuáticas. A continuación, en la **Figura 1** mostramos el árbol de causas y efectos que está relacionado a la adaptación de algoritmo criptográfico basado en curvas elípticas ECIES detallando las causas y efectos que este planteamiento puede llegar a obtener.

Figura 1.

Árbol de causas y efectos enfocado a la obtención de las causas y los efectos que pueden producir una falta de adaptación de algoritmos criptográficos.



Se pone especial atención hacia a las vulnerabilidades y riesgos que pueden llevar estos sistemas en cuestión de mantener su entorno seguro y fuera de los ataques de los atacantes.

Como se pudo observar en la **Figura 1**, la mayor parte de la problemática va a ser enfocada a la Falta de un algoritmo de encriptación para las redes inalámbricas subacuáticas (UWSN) la cual enfocándonos en las causas se denota que la principal barrera que hay que superar en la transmisión segura de los datos y los problemas que pueden tener por las atenuaciones.

A medida que aumenta la dependencia (Kebapci et al., 2022; Lloret Mauri et al., 2015)estas redes para aplicaciones críticas como la exploración de recursos, la vigilancia y la prevención de desastres, es fundamental desarrollar mecanismos de seguridad eficientes y confiables. Para(Cong et al., 2010) la encriptación emerge como una necesidad crucial para los sistemas criptográficos en redes inalámbricas subacuáticas, ya que proporciona una capa adicional de seguridad que protege la confidencialidad y la integridad de los datos transmitidos.

Se investigará la capacidad de la encriptación para mitigar los riesgos de interceptación y manipulación de datos, junto con el desarrollo continuo de algoritmos y protocolos adaptados específicamente a las características del medio subacuático, demuestra su importancia en la seguridad de las redes inalámbricas subacuáticas. Por lo tanto, según detalla (Louza & Jesus, 2022) es crucial examinar cuidadosamente la viabilidad de la encriptación en entornos submarinos y desarrollar soluciones robustas que satisfagan las demandas de seguridad de las aplicaciones críticas en estos contextos.

Por lo tanto, se necesita un estudio basado a la adaptación de algoritomos criptográficos a través de curvas elípticas en las redes de comunicaciones inalámbricas submarinas para identificar y abordar los desafíos clave que enfrentan estas redes en términos de protección de datos y garantía de la integridad de las comunicaciones.

1.3. Objetivos

1.3.1. Objetivo General

Adaptar el sistema criptográfico ECIES a las redes UWSN, con el propósito de evaluar y recopilar información de su desempeño en entornos submarinos, utilizando el simulador OMNeT++.

1.3.2. Objetivos Específicos

- Realizar una investigación sobre las características y desafíos de las redes inalámbricas subacuáticas para comprender su funcionamiento y posibles aplicaciones.
- Determinar los requerimientos necesarios para simular y adaptar del algoritmo de encriptación ECIES enfocándose a las redes UWSN en OMNeT++, para entornos submarinos.
- Implementar un escenario de simulación específico del algoritmo ECIES adaptado a una red inalámbrica subacuática, conforme a los requisitos establecidos, con el fin de

evaluar su eficacia en la transmisión segura de datos bajo condiciones submarinas simuladas, mediante el uso de OMNeT++.

- Evaluar el desempeño del algoritmo ECIES en las redes UWSN, con el objetivo de proporcionar una comprensión detallada de su comportamiento y destacar los resultados obtenidos en términos de seguridad y eficiencia de transmisión de datos en entornos submarinos.

1.4. Alcance

El presente trabajo de investigación se centrará en la adaptación del algoritmo criptográfico basado en curvas elípticas conocido como ECIES (Elliptic Curve Integrated Encryption Scheme), aplicado específicamente a las Redes de Sensores Inalámbricos Subacuáticos (UWSN). El alcance se limita a una exploración técnica profunda, donde se investigarán y considerarán las características específicas del medio subacuático, como la alta atenuación acústica, los largos tiempos de propagación, el bajo ancho de banda disponible y el consumo energético restringido en estos entornos.

Se realizará una detallada revisión bibliográfica actualizada, con el propósito de identificar claramente los desafíos tecnológicos más recientes relacionados con las redes inalámbricas subacuáticas. Este análisis permitirá sustentar adecuadamente la necesidad y viabilidad de adaptar ECIES, considerando factores cruciales como la eficiencia energética, la seguridad criptográfica y la estabilidad operativa frente a condiciones ambientales adversas, tales como variaciones en temperatura, salinidad y presión hidrostática.

El trabajo contempla específicamente la implementación del algoritmo ECIES adaptado en simulaciones computacionales desarrolladas utilizando la herramienta especializada OMNeT++. Se diseñarán escenarios de simulación representativos del ambiente marino, cubriendo diversas situaciones y condiciones realistas, como diferentes profundidades,

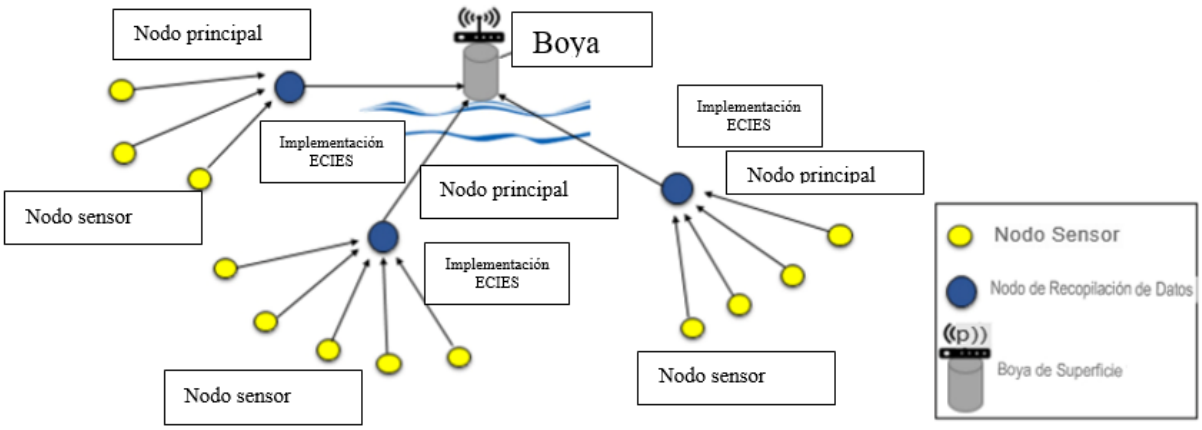
distribución espacial de los sensores y posibles interferencias producidas por corrientes oceánicas, turbulencias y ruido ambiental.

Las simulaciones tendrán como objetivo evaluar en detalle el desempeño de la versión adaptada de ECIES como se puede ver en la **Figura 2**. Las métricas consideradas para la evaluación serán principalmente: la seguridad criptográfica proporcionada (integridad, confidencialidad y autenticación), eficiencia energética de la transmisión y recepción de datos, así como la fiabilidad y robustez del algoritmo frente a las particularidades operativas del entorno submarino simulado.

Finalmente, cabe destacar que el alcance del presente estudio se limita estrictamente al ámbito simulado. No se contemplan pruebas físicas o implementaciones experimentales en ambientes subacuáticos reales. Las conclusiones obtenidas serán únicamente aplicables a los contextos simulados definidos en OMNeT++, proporcionando una base sólida y metodológicamente válida para futuras investigaciones experimentales o implementaciones prácticas en entornos reales.

Figura 2

Arquitectura general de una red de sensores inalámbricos subacuáticos.



Nota: Se pueden observar los diferentes nodos de sensores y las boyas en general. Se va a adaptar un algoritmo de curva elíptica para evaluar la comunicación entre ellos.

1.5. Justificación

La investigación propuesta aborda desafíos críticos en el campo de las comunicaciones submarinas, dado que más del 75% de la superficie de la Tierra está cubierta por agua en forma de océanos (Ali et al., 2020) A pesar de su vastedad, el mundo submarino sigue siendo mayormente inexplorado, y las redes de sensores inalámbricos subacuáticos (UWSN) desempeñan un papel crucial en la observación y estudio de la vida marina, la contaminación del agua, la exploración de plataformas de petróleo y gas, entre otras actividades (Cong et al., 2010).

Sin embargo, estas redes enfrentan limitaciones únicas debido a las condiciones ambientales submarinas, como alta atenuación, dispersión y absorción de señales. A diferencia de las redes de sensores terrestres, las UWSN operan en un entorno con recursos limitados y condiciones adversas, lo que plantea desafíos adicionales en términos de velocidad de transmisión, alcance y consumo energético (Marine Technology Society et al., 2012)& (Sathish et al., 2022).

La seguridad de los datos en las UWSN es crítica debido a la sensibilidad de la información transmitida y su vulnerabilidad ante posibles ataques cibernéticos que podrían comprometer la integridad, confidencialidad y disponibilidad de los datos (Lloret Mauri et al., 2015)& (Sathish et al., 2022). Por lo tanto, es fundamental garantizar la protección de la información en estos entornos submarinos.

Desde el punto de vista tecnológico, la implementación del algoritmo criptográfico ECIES ofrece una solución eficaz para abordar estos desafíos de seguridad en las UWSN. Este algoritmo proporciona un alto nivel de seguridad con claves más cortas, lo que lo hace adecuado para entornos con recursos limitados como las redes submarinas (Marine Technology Society et al., 2012; Technische Universität Wien et al., 2016).

La investigación propuesta busca evaluar la viabilidad y efectividad de ECIES en entornos submarinos mediante simulaciones en OMNeT++. Esto tiene el potencial de contribuir al desarrollo de soluciones prácticas para mejorar la seguridad de las comunicaciones submarinas. Además, como analizamos la investigación de (S.-H. Chang et al., 2015), la investigación tiene una dimensión teórica importante al abordar el desafío de aplicar un algoritmo de cifrado avanzado en un entorno submarino único y complejo.

Y de acuerdo con (Ali et al., 2020; IEEE Computer Society. et al., 2013) la utilización de simulaciones en OMNeT++ como metodología permite realizar experimentos controlados en un entorno virtual, proporcionando insights valiosos sobre el rendimiento y la efectividad de ECIES en condiciones submarinas. Esto contribuye al desarrollo de métodos y técnicas para abordar los desafíos de seguridad en las UWSN de manera eficiente y confiable.

Capítulo II: Fundamento Teórico

Este capítulo reúne los conceptos que sustentan la adaptación de un esquema criptográfico basado en curvas elípticas para redes inalámbricas subacuáticas (UWSN). Primero se describen las UWSN y las características del medio subacuático que condicionan la comunicación. Luego se presentan los aspectos de seguridad más relevantes (amenazas, requisitos y mecanismos típicos) y, finalmente, se introduce la criptografía de curva elíptica y el esquema ECIES como base de la propuesta. El capítulo cierra con una breve revisión de la simulación en OMNeT++ y la metodología Kanban utilizada para organizar el trabajo.

El objetivo principal es observar que en el entorno subacuático el canal suele ser lento, ruidoso y con alto retardo. Por eso, cualquier mecanismo de seguridad debe ser fuerte, pero también práctico: que use pocos mensajes, poco ancho de banda y un costo computacional razonable para nodos con energía limitada (Akyildiz et al., 2005a)

2.1. Redes Inalámbricas Subacuáticas (UWSN)

Las redes inalámbricas subacuáticas de sensores (Underwater Wireless Sensor Networks, UWSN) están formadas por nodos sensores y nodos de apoyo desplegados en un medio acuático para observar, medir y reportar variables de interés (por ejemplo, temperatura, salinidad, presión, turbidez o parámetros asociados a infraestructura submarina) y los envían hacia un nodo recolector (gateway/sink) para su procesamiento o transmisión hacia la superficie (Heidemann et al., 2012). En términos generales, las UWSN persiguen objetivos equivalentes a las redes de sensores terrestres; sin embargo, operan sobre un medio de comunicación significativamente más exigente (Xie et al., 2021a). Revisiones y encuestas recientes resumen aplicaciones y retos en monitoreo ambiental, vigilancia, exploración y gestión de riesgos (Felemban et al., 2015).

Estas redes habilitan aplicaciones de monitoreo ambiental, exploración y vigilancia de activos marinos; sin embargo, a diferencia de las redes terrestres, su diseño está condicionado por restricciones severas del canal subacuático, la dificultad de mantenimiento y el alto costo energético de la comunicación (Ali et al., 2020).

En UWSN, la conectividad se logra principalmente mediante enlaces acústicos, aunque en escenarios específicos también se consideran enlaces ópticos (luz visible) y, a distancias muy cortas o aplicaciones puntuales, soluciones basadas en radiofrecuencia. La selección del medio de transmisión afecta directamente la topología, el consumo energético, la confiabilidad del enlace y, en consecuencia, el diseño de los mecanismos criptográficos y de seguridad de red.

2.1.1 Introducción a las UWSN

Una UWSN suele incluir: (a) nodos sensores, (b) nodos de retransmisión (relays) cuando se requiere multisalto, (c) un nodo sink o gateway, y (d) un enlace hacia una estación en superficie. En algunos escenarios se incorporan vehículos autónomos (AUV/UUV) como nodos móviles para recolección o inspección como se puede observar en la **Tabla 1** (Chaudhary et al., 2023).

Tabla 1

Componentes típicos en una UWSN y su función.

Componente	**Rol en la red**	**Notas prácticas**
Nodo sensor	Mide variables del entorno y genera paquetes de datos.	Limitado en energía y procesamiento; suele dormir la mayor parte del tiempo.
Nodo relay	Reenvía tráfico para extender cobertura (multisalto).	Aumenta alcance, pero añade retardo y consumo.

Sink / Gateway	Recibe datos y coordina la red; puede ser punto de agregación.	Suele tener más capacidad de cómputo y batería.
Estación en superficie	Almacena/visualiza datos; conecta con redes terrestres.	Enlace con mayor disponibilidad de energía.
AUV/UUV (opcional)	Nodo móvil para recolección o apoyo de conectividad.	Útil cuando la red es dispersa o el canal es muy variable.

Nota. El diseño exacto depende del escenario (densidad, alcance, movilidad y objetivos de monitoreo).

2.1.2 Medios de comunicación subacuática

A diferencia de las redes terrestres, bajo el agua la radiofrecuencia (RF) se atenúa con rapidez. Por eso, la mayoría de UWSN se apoya en comunicación acústica para distancias medias y largas, mientras que la comunicación óptica (por ejemplo, luz visible) se utiliza cuando se necesita alto ancho de banda en distancias cortas y con línea de vista (Y. Chang et al., 2015; Pal et al., 2022).

En términos prácticos: La **Tabla 2** nos muestra que la acústica llega más lejos, pero es lenta; la óptica es rápida, pero exige condiciones más estrictas (turbidez, alineación y rango corto) (Ali et al., 2019).

Tabla 2

Comparación general de medios de comunicación en UWSN.

Medio	Rango típico	Ventajas	Limitaciones
Acústico	Medio–largo	Mejor cobertura subacuática; adecuado para redes dispersas.	Bajo ancho de banda, alto retardo, multipropagación y ruido.
Óptico / VLC	Corto	Alto ancho de banda y baja latencia.	Requiere línea de vista; sensible a turbidez y alineación.
Electromagnético (RF)	Muy corto	Puede ser útil en casos específicos (agua poco profunda, proximidad).	Alta atenuación en agua; rango muy limitado.

Nota. Comparación cualitativa basada en revisiones de comunicación subacuática (Pal et al., 2022).

2.1.3 Topologías y enrutamiento en UWSN

La topología de una UWSN depende de la aplicación. En despliegues pequeños es posible una comunicación directa (un salto) hacia el sink, pero lo más común es usar multisalto para ampliar cobertura. En estos casos, el enrutamiento debe considerar que la posición de los nodos puede variar por corrientes y que el canal cambia con el tiempo (Ahmed et al., 2017).

Existen propuestas de enrutamiento que aprovechan información del entorno. Por ejemplo, Pressure Routing utiliza presión/profundidad para guiar el reenvío sin necesidad de

una tabla de rutas (Lee & Kim, 2010). Otras propuestas como Mobicast buscan soportar difusión dirigida considerando movilidad (Liu et al., 2008).

Cuando la red crece, se suele recurrir a estrategias de agrupamiento (clustering). La idea es formar grupos pequeños con un nodo cabecera (cluster head) para reducir tráfico y ahorrar energía (M. K. Khan & Alghathbar, 2015). Adicionalmente, se han propuesto esquemas de ruteo que consideran energía disponible o incluso recolección de energía (energy harvesting) para extender vida útil (Ahmed et al., 2017).

Finalmente, para aplicaciones que requieren conocer la ubicación, es común apoyarse en algoritmos de localización. En UWSN esto es un reto por el retardo y la variabilidad del canal, por lo que se han propuesto enfoques específicos y revisiones dedicadas al tema (Luo et al., 2023).

2.1.4 Parámetros críticos del canal acústico

Aunque existen alternativas ópticas y de radiofrecuencia, en la mayoría de los despliegues de UWSN la portadora dominante es la acústica debido a su alcance. Por ello, es clave comprender los parámetros del canal acústico que afectan la confiabilidad, el consumo energético y la temporización de los protocolos (Ali et al., 2019; Xie et al., 2021a)

2.1.4.1 Atenuación y pérdidas de transmisión

La atenuación acústica incrementa con la frecuencia, lo que crea un compromiso directo entre alcance y tasa de datos. A frecuencias altas se pueden alcanzar mayores bitrates, pero el alcance se reduce significativamente. Una aproximación común para estimar la absorción (en dB/km) es la ecuación conocida como fórmula de Thorp.

Ecuación 1

Coeficiente de absorción acústica en agua de mar (modelo de Thorp)

$$\alpha(f) = \frac{0.11f^2}{1+f^2} + \frac{44f^2}{4100+f^2} + 2.75 \times 10^{-4} f^2 + 0.003 \qquad (1)$$

donde:

$\alpha(f)$: coeficiente de absorción (dB/km).

f: frecuencia de la señal (kHz).

En modelos de enlace, la pérdida total de transmisión combina la absorción con pérdidas por dispersión geométrica y efectos de multitrayecto. En simulación, estas pérdidas suelen parametrizarse mediante modelos de propagación y ruido que dependen del escenario (profundidad, salinidad, temperatura y actividad ambiental) (Shin & Park, 2008).

2.1.4.2 Velocidad de propagación del sonido

La velocidad del sonido en el agua no es constante; varía principalmente con la temperatura, la salinidad y la presión (o profundidad). Este parámetro influye de forma directa en el cálculo de retardo de propagación, en la sincronización y en el diseño de ventanas de tiempo para autenticación y control de sesiones. Un modelo frecuentemente empleado en simulación del canal, y utilizado en implementaciones basadas en OMNeT++, es el modelo de Mackenzie (Shin & Park, 2008).

Ecuación 2

Velocidad del sonido en el océano (ecuación de Mackenzie)

$$c(T,S,z) = 1448.96 + 4.591T - 0.05304T^2 + 0.0002374T^3 + 1.34(S-35) + 0.0163z \\ -1.675\times10^{-7}z^2 - 0.01025T(S-35) - 7.139\times10^{-13}Tz^3 \qquad (2)$$

donde:

c: velocidad del sonido en el agua (m/s).

T: temperatura del agua $(°C)$.

S: salinidad (ppt).

z: profundidad (m).

2.1.4.3 Retardo de propagación y sincronización

En medios acústicos, el retardo de propagación es considerablemente mayor que en enlaces de radiofrecuencia en aire, debido a que la velocidad de propagación es del orden de $1.5 \times 10^3 \mathrm{m/s}$. Este retardo impacta los tiempos de espera (timeouts), el control de retransmisiones, la sincronización y los mecanismos anti-replay en protocolos de seguridad, especialmente cuando se utilizan desafíos/respuestas o intercambio de claves con múltiples mensajes (Xie et al., 2021; Yu & Lee, 2023).

Ecuación 3

Retardo de propagación acústica

$$t_p = d/c \tag{3}$$

donde:

t_p: tiempo de propagación (s).

d: distancia entre emisor y receptor (m).

c: velocidad del sonido en el agua (m/s).

2.1.4.4 Ancho de banda y capacidad del canal

El ancho de banda utilizable en enlaces acústicos suele ser reducido y depende de la distancia y del rango de frecuencias empleadas. Esto limita la tasa de bits efectiva y obliga a optimizar el tamaño de mensajes de control y seguridad. En términos teóricos, la capacidad máxima del canal puede aproximarse mediante la relación de Shannon-Hartley (Shannon, 1948):

Ecuación 4

Capacidad de canal (Shannon–Hartley)

$$C = B \cdot log^2(1 + SNR) \tag{4}$$

donde:

C: capacidad del canal (bps).

B: ancho de banda (Hz).

SNR: relación señal a ruido.

En la práctica, las UWSN operan por debajo del límite teórico debido a ruido ambiental, interferencias, desvanecimiento y multitrayecto. Por ello en la **Tabla 3** observamos que los diseños criptográficos deben asumir pérdidas, retrasos y reintentos, sin depender de intercambios largos de mensajes (Ali et al., 2019).

Tabla 3

Parámetros críticos del canal acústico y su impacto en UWSN.

Parámetro	**Qué significa (en simple)**	**Impacto directo**
Retardo de propagación	Tiempo que tarda un paquete en viajar por el agua.	Aumenta la latencia y complica protocolos con muchos mensajes.
Ancho de banda	Cuánta información cabe por segundo.	Limita tamaño de tramas; obliga a reducir overhead criptográfico.
Atenuación/absorción	La señal se debilita con distancia y frecuencia.	Reduce alcance; aumenta necesidad de retransmisiones o relays.
Multipropagación (multipath)	La señal llega por varios caminos con ecos.	Provoca interferencia y errores; afecta la tasa de entrega.
Ruido (ambiental y de equipos)	Perturbaciones naturales o artificiales.	Reduce SNR y capacidad; incrementa BER.

Nota. Resumen basado en los desafíos clásicos de UWSN (Akyildiz et al., 2005a) y revisiones recientes (Ali et al., 2019).

2.1.5 Parámetros críticos en comunicaciones subacuáticas

El entorno subacuático representa uno de los medios más complejos para la transmisión de datos, debido a que múltiples factores ambientales modifican de forma significativa las propiedades del canal acústico. Entre estos factores, los más determinantes son la salinidad, la temperatura y la presión. Estas variables influyen directamente sobre la velocidad de propagación, la atenuación acústica, la densidad del agua y, en general, en el desempeño de las redes de sensores inalámbricos subacuáticos (UWSN).

En síntesis, el canal acústico subacuático impone alto retardo, ancho de banda limitado y pérdidas dependientes del entorno; en consecuencia, un mecanismo de seguridad viable debe reducir rondas de intercambio, minimizar bytes transmitidos y controlar el costo computacional. Con esta base, la siguiente sección formaliza los requisitos de seguridad y amenazas típicas en UWSN.

2.1.5.1 Salinidad

La salinidad del agua, entendida como la concentración de sales disueltas, afecta la densidad del medio y, por ende, la velocidad con la que se propagan las ondas acústicas. A medida que la salinidad incrementa, la densidad del agua aumenta, provocando una ligera aceleración en la propagación de las ondas. En condiciones oceánicas típicas, la salinidad varía entre 33 y 37 partes por mil (ppt), generando diferencias en la velocidad de hasta $4\,\mathrm{m/s}$ (Jensen et al., 2011). Además, la salinidad puede influir en la absorción acústica mediante el aumento de la conductividad del medio.

2.1.5.2 Temperatura

La temperatura es el factor con mayor impacto directo sobre la velocidad del sonido en el agua. Un incremento de $1\,°C$ puede provocar una aceleración de aproximadamente 4 a $5\,\mathrm{m/s}$

en la propagación acústica. Este efecto se debe a la reducción de la densidad y al aumento en la elasticidad del agua a temperaturas más elevadas (Ainslie, 2010). Como consecuencia, en zonas superficiales expuestas al sol o en regiones tropicales, la velocidad del sonido puede ser sustancialmente mayor, alterando la sincronización entre nodos y afectando los tiempos de retardo en UWSN.

2.1.5.3 Presión

La presión, estrechamente relacionada con la profundidad, también tiene un impacto considerable en la velocidad de propagación. A mayor profundidad, el agua se comprime ligeramente, aumentando la densidad del medio y, en consecuencia, la velocidad del sonido. Este fenómeno es especialmente importante en aplicaciones de sensores a gran profundidad, como redes instaladas en fondos oceánicos o zonas de exploración petrolera (Urick, 1983). La presión también incide en la eficiencia de los transductores acústicos utilizados en los nodos, afectando su rango y fidelidad de transmisión.

En conjunto, estos tres parámetros constituyen la base para calcular la velocidad del sonido de forma precisa, como lo hace la ecuación de Mackenzie (ver Sección 2.1.3 Topologías y enrutamiento en UWSN), y son imprescindibles en modelos de simulación física de redes UWSN. El desconocimiento o la omisión de estos factores puede conducir a simulaciones poco realistas y diseños de red ineficaces.

2.2. Seguridad en redes inalámbricas subacuáticas

La seguridad en UWSN busca proteger la información y la operación de la red frente a atacantes capaces de escuchar, modificar, inyectar tráfico o degradar el servicio. En un escenario real, un atacante puede escuchar el canal, inyectar paquetes, suplantar nodos o incluso capturar un nodo físicamente. Por esa razón, la seguridad en redes subacuáticas se estudia como un problema integral que incluye confidencialidad, integridad, autenticación y disponibilidad (Saeed et al., 2023).

2.2.1 Requisitos básicos de seguridad

De forma resumida, una UWSN segura debería cumplir al menos con: (a) confidencialidad, para evitar que terceros lean los datos; (b) integridad, para detectar modificaciones; (c) autenticación, para verificar quién envía cada mensaje; y (d) frescura (anti-replay), para impedir que mensajes antiguos se reutilicen como ataques. En redes subacuáticas, además, la disponibilidad es crítica porque las retransmisiones son costosas (Domingo, 2011; Dong & Liu, 2011).

2.2.2 Amenazas frecuentes en UWSN

Las UWSN operan en un entorno abierto y difícil de proteger físicamente, donde la comunicación acústica facilita la escucha del canal y la interferencia deliberada. En los últimos años, el interés por la seguridad marítima y la protección de infraestructura submarina crítica (cables y ductos) ha incrementado la atención sobre amenazas híbridas en el dominio subacuático, combinando sabotaje físico con acciones cibernéticas sobre los sistemas de monitoreo (European Commission, 2024, 2025).

Desde un punto de vista operativo, el atacante puede clasificarse como: (i) pasivo (listener), que observa el tráfico para extraer información o realizar análisis de patrones; (ii) activo, que inyecta, modifica o suprime paquetes para alterar el funcionamiento de la red; y (iii) físico, con capacidad de capturar o manipular nodos, comprometiendo claves y parámetros internos. Estas capacidades suelen combinarse en ataques multi-etapa (por ejemplo, captura → suplantación → replay).

Las amenazas emergentes se relacionan con (a) interferencia y jamming reactivo/inteligente, que busca degradar disponibilidad con gasto energético mínimo; (b) spoofing de localización/profundidad y ataques a sincronización temporal, que afectan el enrutamiento y los mecanismos de coordinación; y (c) compromisos en el gateway o el enlace de retorno, que abren un vector de ataque “cross-domain” desde redes superficiales hacia el

dominio subacuático. La **Tabla 4** organiza amenazas clásicas y actuales, junto con controles típicos de mitigación. (Alharbi & Ibrahim, 2026; Mertens et al., 2023).

Tabla 4

Amenazas clásicas y emergentes en UWSN y controles recomendados.

Amenaza	**Descripción (efecto típico)**	**Controles/mitigación (ejemplos)**
Escucha pasiva y análisis de tráfico (listener)	Intercepción del canal para inferir contenido o patrones (quién habla, cuándo, cuánto).	Cifrado del payload; ofuscación de metadatos cuando sea posible; padding/aleatorización de tráfico en casos críticos.
Suplantación/impersonación	Un nodo malicioso se hace pasar por un nodo legítimo para acceder a la red o desviar tráfico.	Autenticación (firmas o MAC), gestión de identidades, listas de control y validación de JOIN.
Replay	Reenvío de mensajes válidos capturados previamente para forzar acciones repetidas o confundir estados.	Nonces/contadores/ventanas anti-replay; verificación de frescura antes de procesar.
Jamming e interferencia reactiva/inteligente	Interferencia deliberada del canal (constante o reactiva) para degradar	Detección de interferencia, técnicas anti-jamming (salto de frecuencia/espacio), rutas

	disponibilidad y agotar energía.	alternativas, control de potencia y tolerancia a fallos.
Ataques a sincronización y localización (p. ej., depth-spoofing/ToA spoofing)	Manipulación de señales/medidas que induce errores de tiempo o posición, afectando coordinación y enrutamiento.	Balizas autenticadas, chequeos de consistencia, fusión de sensores, validación cruzada y políticas de confianza.
Inyección/alteración de datos (False Data Injection)	Inserción de lecturas o mensajes falsos, o modificación del contenido para engañar a la aplicación.	Firmas/validación de integridad extremo a extremo; controles de plausibilidad; redundancia y votación.
Ataques de enrutamiento (sinkhole, selective forwarding, blackhole)	Atracción y/o descarte selectivo de paquetes para degradar PDR o sesgar rutas.	Ruteo multipath, mecanismos de confianza, detección de anomalías y monitoreo de tasa de entrega.
Wormhole	Creación de un “túnel” de baja latencia entre dos puntos para distorsionar topología y rutas.	Leashes (temporales/geográficos) y chequeos RTT; validación de vecinos; detección por inconsistencias topológicas.
Sybil	Un nodo adopta múltiples identidades para influir en	Identidades fuertes (certificados/credenciales), verificación por desafío-

	votaciones, ruteo o asignación de recursos.	respuesta, límites por recurso.
DoS por flooding / saturación de control	Inyección masiva de tráfico para congestionar el canal o forzar procesamiento excesivo.	Rate limiting, priorización, verificación temprana (verify→decrypt), rechazo por política y listas de bloqueo.
Compromiso de gateway / ataques cross-domain	Intrusión a componentes de superficie (gateway/backhaul) para manipular comandos o exfiltrar datos.	Endurecimiento del gateway, autenticación mutua, segmentación, auditoría y actualización segura.
Captura física / sabotaje de infraestructura (nodos, cables, sensores)	Manipulación, extracción o daño físico; recuperación de claves; interrupción del servicio.	Protección física y redundancia; revocación/rotación de claves; monitoreo de integridad y detección de tampering.

Nota. Síntesis basada en literatura de seguridad para UWSN, con énfasis en jamming inteligente y spoofing de localización/profundidad, y en lineamientos recientes sobre protección de infraestructura submarina crítica.

2.2.3 Autenticación y control de acceso

En UWSN, autenticar no significa solo “saber quién es quién”; significa también evitar gasto de energía por paquetes falsos. Por eso se han propuesto esquemas de autenticación extremo a extremo y protocolos ligeros que reducen mensajes de control. Por

ejemplo, se han reportado enfoques de autenticación extremo a extremo (Souza & Vieira, 2013) y protocolos basados en tickets para disminuir overhead (Yun & Park, 2016). Cuando el atacante intenta multiplicar identidades (Sybil), se han planteado mecanismos específicos para validar identidad y comportamiento del nodo (A. A. Islam & Taher, 2021). Además, modelos de confianza que aprovechan el canal o el historial de comunicación pueden ayudar a detectar nodos sospechosos (Signori & Casari, 2022).

En escenarios con movilidad (por ejemplo, nodos arrastrados por corrientes), la autenticación y la protección de la comunicación suelen requerir aún más cuidado, porque el estado de la red cambia con frecuencia. Existen propuestas específicas para comunicación segura en UWSN móviles ((Das & Thampi, 2015).

También se han explorado arquitecturas basadas en SDN (Software-Defined Networking) para separar control y datos, y centralizar decisiones de autenticación de forma escalable (Liu et al., 2008). En esta tesis, estas ideas se consideran como contexto, mientras que el mecanismo propuesto se centra en cifrado y establecimiento de clave eficiente.

2.2.4 Gestión de claves y equilibrio energía–seguridad

La gestión de claves es uno de los puntos más sensibles. Si una red debe intercambiar claves con muchos nodos, el tiempo de convergencia puede crecer y el canal puede saturarse. Por ello, existen trabajos que analizan el desempeño de establecimiento de claves y el costo de autenticación (Krivokapić & Vukotic, 2023). También se han propuesto enfoques de optimización conjunta entre energía y seguridad en redes subacuáticas, evidenciando que un esquema más seguro puede incrementar consumo y latencia si no se diseña con cuidado (M. M. Islam et al., 2020).

En proyectos aplicados se reporta que la seguridad práctica debe ser compatible con las limitaciones del canal y el hardware, y que las decisiones de diseño deben medirse con simulación o pruebas controladas (Casari & Zorzi, 2022).

2.2.5 Consideraciones de escalabilidad

Cuando la red crece (decenas o cientos de nodos), intercambiar material criptográfico entre todos los dispositivos puede volver lenta la puesta en marcha de la red. Una estrategia común es organizar la red en clusters: los nodos de cada cluster se coordinan con su cabecera y solo ciertas cabeceras negocian con el sink. Esto reduce mensajes globales y facilita el control de claves (M. K. Khan & Alghathbar, 2015).

A partir de los requisitos (confidencialidad, integridad, autenticación y frescura) y de las amenazas identificadas, el esquema criptográfico seleccionado debe: (i) reducir rondas de intercambio, (ii) minimizar bytes adicionales, (iii) permitir verificación temprana para descartar tráfico inválido sin gastar energía y (iv) escalar a decenas o cientos de nodos. Bajo estas condiciones, un enfoque híbrido basado en curvas elípticas resulta adecuado para UWSN.

2.3. Criptografía basada en curvas elípticas y esquema ECIES

En redes con restricciones de energía y ancho de banda, suele ser más eficiente combinar criptografía asimétrica y simétrica. La asimétrica se usa para acordar una clave de sesión, y la simétrica para cifrar los datos. A esto se le conoce como enfoque híbrido, y es el patrón que siguen esquemas como ECIES (International Organization for Standardization & International Electrotechnical Commission, 2006). La **Tabla 5** presenta una comparación de enfoques criptográficos relevantes para UWSN, destacando su adecuación en escenarios con recursos limitados.

Tabla 5

Comparación de enfoques criptográficos en UWSN

Enfoque	Ventaja principal	Desventaja principal	Uso típico en UWSN
Simétrico (p. ej., AES)	Muy rápido y eficiente.	Requiere una clave compartida previa.	Cifrado de datos una vez establecida la clave.
Asimétrico (p. ej., ECC)	Facilita intercambio de claves y autenticación.	Más costoso que simétrico.	Establecimiento de claves; firmas o autenticación.
Híbrido (ECC + AES)	Equilibra seguridad y eficiencia.	Debe diseñarse bien el protocolo.	ECIES: ECC para clave y AES para datos.

Nota. En UWSN se favorece el enfoque híbrido para reducir overhead y mantener seguridad (Hankerson et al., 2004).

2.3.1 Fundamentos de criptografía de curva elíptica (ECC)

La criptografía de curva elíptica (ECC) se basa en operaciones matemáticas sobre puntos de una curva elíptica definida sobre un campo finito. La seguridad práctica se apoya en la dificultad del problema del logaritmo discreto en curvas elípticas (ECDLP). En términos de implementación, ECC permite alcanzar niveles de seguridad altos con tamaños de clave relativamente pequeños (Hankerson et al., 2004) (Standards for Efficient Cryptography Group (SECG), 2009).

En ECC, las claves se derivan de operaciones de multiplicación escalar de puntos sobre una curva elíptica definida en un campo finito. A grandes rasgos, una clave privada es un escalar k y la clave pública es el punto $\mathrm{K} = \mathrm{k} \cdot G$, donde G es un punto generador acordado (parámetros del dominio). La operación dominante es la multiplicación escalar, que

debe implementarse de forma eficiente y resistente a ataques de canal lateral cuando el hardware lo requiera.

Una forma estándar de representar la curva (en un campo primo) es:

Ecuación 5

Ecuación general de curva elíptica sobre campo primo

$$y^2 = x^3 + ax + b(modp) \quad (5)$$

Para UWSN, dos consideraciones prácticas son especialmente relevantes: (a) la reducción del tamaño de mensajes mediante representación comprimida de puntos (point compression) para disminuir bytes transmitidos; y (b) el uso de claves efímeras para proporcionar secreto hacia adelante (forward secrecy) en caso de compromiso posterior de claves a largo plazo. Estas decisiones impactan directamente la sobrecarga de comunicación, que es un recurso crítico en el medio subacuático (Yu & Lee, 2023).

2.3.2 Funcionamiento técnico de ECIES

En esta tesis se adopta un perfil ECIES-based donde el acuerdo de secreto se realiza con ECDH sobre P-256, la confidencialidad con AES-CTR, y la integridad/autenticación mediante firma ECDSA (SHA-256) sobre el ciphertext y metadatos. Este diseño habilita el flujo “verificar → descifrar”, descartando mensajes inválidos antes de consumir recursos en descifrado.

El algoritmo ECIES opera bajo un esquema híbrido que combina criptografía de clave pública, cifrado simétrico y mecanismos de autenticación para garantizar la confidencialidad e integridad de los datos. Su funcionamiento puede dividirse conceptualmente en dos etapas: la fase de cifrado, realizada por el emisor, y la fase de descifrado, llevada a cabo por el receptor. Cada fase implica una serie de pasos criptográficos interrelacionados, los cuales son críticos

para el establecimiento de una comunicación segura, especialmente en redes subacuáticas donde los recursos son limitados y las condiciones del canal son hostiles.

A nivel conceptual, ECIES se compone de: (1) un acuerdo de secreto compartido (por ejemplo, ECDH), (2) una función de derivación de clave (KDF), (3) un cifrado simétrico (AES u otro) y (4) un mecanismo de autenticación/integridad (National Institute of Standards & Technology, 2001, 2023) En esta tesis, la autenticación e integridad del mensaje se implementa mediante firma digital ECDSA (SHA-256) sobre el ciphertext (y metadatos asociados), permitiendo verificar el origen y detectar alteraciones antes del descifrado. La **Tabla 6** sintetiza los componentes del esquema ECIES y el propósito de cada uno dentro del perfil adoptado.

Tabla 6

Componentes de ECIES y propósito

Componente	**¿Qué hace?**	**¿Por qué es importante en UWSN?**
ECDH (secreto compartido)	Permite que emisor y receptor deriven la misma clave sin compartirla explícitamente.	Evita enviar la clave de sesión por el canal.
KDF	Convierte el secreto compartido en claves útiles (cifrado y autenticación).	Permite separar claves y mejorar robustez del diseño.
AES (cifrado simétrico)	Cifra los datos de aplicación con alta eficiencia.	Menor costo computacional para grandes volúmenes de datos.

Firma ECDSA (integridad/autenticación)	Genera una firma sobre el ciphertext (y metadatos) que permite verificar autenticidad del emisor e integridad del mensaje.	Evita aceptar mensajes alterados o falsificados sin depender de un esquema AEAD; la verificación se realiza antes del descifrado.
Clave pública del emisor (operativa en la simulación)	Clave temporal que el emisor adjunta al mensaje.	Clave pública comprimida que el emisor adjunta al paquete para permitir (i) la derivación del secreto ECDH en el receptor y (ii) la verificación de la firma ECDSA, bajo el modelo experimental controlado.

Nota. Resumen conceptual basado en (Standards for Efficient Cryptography Group (SECG), 2009) y en la estandarización de esquemas asimétricos (International Organization for Standardization & International Electrotechnical Commission, 2006).

2.3.2.1 Curva estándar secp256r1 (P-256)

La secp256r1, designada asimismo P-256 en el estándar FIPS 186-5, es la curva elíptica de 256 bits más adoptada en implementaciones de ECIES debido a que combina ≈128 bits de seguridad con un soporte extenso en bibliotecas criptográficas modernas y módulos de hardware seguro. Definida sobre el cuerpo primo $\mathrm{p} = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$, la curva emplea el polinomio cúbico $y\text{^}2 = x\text{^}3 - 3\mathrm{x} + \mathrm{b}$ ($\mathrm{cona} = -3$ para acelerar la aritmética) y un orden del grupo basado en el primo de Cofactor 1, características que facilitan implementaciones resistentes a ataques de canal lateral y algoritmos de verificación por firmas ECDSA. Estos atributos favorecen su selección en entornos restringidos (incluidas las UWSN)

donde se requiere un equilibrio entre consumo energético, compatibilidad con infraestructuras de clave pública existentes y garantías de seguridad a largo plazo (M. Khan & Kim, 2022; NIST, 2023a).

En el enfoque adoptado, el intercambio de secreto se basa en ECDH sobre P-256. El emisor genera un par efímero (d_e, R = d_e·G) y calcula el secreto compartido S = d_e·Q_R usando la clave pública del receptor. A partir de S se deriva una clave simétrica K_enc mediante KDF (SHA-256), utilizada para cifrar el payload con AES en modo CTR y un IV/nonce único por mensaje. Dado que CTR no aporta integridad, el emisor calcula una firma ECDSA (SHA-256) sobre el ciphertext y metadatos mínimos (p. ej., IV/nonce y contador/identificador anti-replay). El receptor recomputa S = d_R·R, deriva K_enc, verifica primero la firma y, solo si es válida, procede al descifrado con AES-CTR.

Ecuación 6

Derivación de clave pública en ECC (multiplicación escalar)

$$S = k * P \tag{6}$$

2.3.3 Ventajas de ECC en redes subacuáticas

La principal ventaja de ECC para UWSN es que reduce el tamaño de material criptográfico que se transmite. Por ejemplo, para niveles de seguridad comparables, una clave ECC suele ser mucho más corta que una clave RSA tradicional, lo que se traduce en menos bytes por mensaje y menos tiempo de ocupación del canal (Hankerson et al., 2004). En un canal acústico donde cada byte cuesta tiempo y energía, esta diferencia importa.

Además, ECC es compatible con esquemas modernos de establecimiento de claves (por ejemplo, ECDH) y con recomendaciones de estándares para intercambio de claves, lo cual facilita una implementación más segura y verificable (Standards for Efficient Cryptography Group (SECG), 2009; International Organization for Standardization & International Electrotechnical Commission, 2006).

2.3.4 AES y modos de operación relevantes en comunicaciones seguras

AES (Advanced Encryption Standard) es el cifrado simétrico más utilizado en sistemas contemporáneos; sin embargo, su seguridad práctica depende del modo de operación y de la correcta gestión de parámetros como IV/nonce. En esquemas híbridos basados en curvas elípticas, como ECIES, AES se emplea para proteger el payload con alta eficiencia, mientras que las primitivas de curva elíptica se utilizan para acordar material de clave y soportar mecanismos de autenticación según el diseño del esquema (International Organization for Standardization & International Electrotechnical Commission, 2006; Standards for Efficient Cryptography Group (SECG), 2009).

En términos generales, algunos modos de operación proveen únicamente confidencialidad (p. ej., CBC y CTR) y, por tanto, requieren un mecanismo adicional para integridad y autenticación; en contraste, los modos AEAD integran cifrado e integridad en una sola construcción. La **Tabla 7** compara modos de operación AES relevantes para comunicaciones seguras, destacando sus propiedades de seguridad y sus implicaciones de implementación en entornos con recursos limitados.

Tabla 7

Comparación de modos AES relevantes para comunicaciones seguras

Modo AES	Propiedad principal	¿Requiere IV/nonce?	Integridad incluida	Riesgo clave si se implementa mal	Comentario en UWSN
CBC	Confidencialidad	Sí (IV)	No	Padding/oráculos; necesidad de MAC/firma	Puede ser más costoso por padding y manejo de bloques

CTR	Confidencialidad	Sí (nonce/contador)	No	Reutilizar (K, nonce) rompe confidencialidad	Eficiente y fácil de instrumentar; exige unicidad estricta
AEAD (p. ej., GCM)	Confidencialidad + integridad	Sí	Sí	Reutilizar nonce puede ser crítico	Útil, pero en esta tesis se separa cifrado vs autenticación

En particular, el modo CTR convierte a AES en un cifrador tipo flujo, lo que resulta atractivo en entornos restringidos: evita dependencias de padding y permite un procesamiento más directo del payload. No obstante, CTR impone una condición crítica: no reutilizar el par (clave, IV/nonce); de lo contrario, se compromete la confidencialidad del esquema (NIST, 2023a).

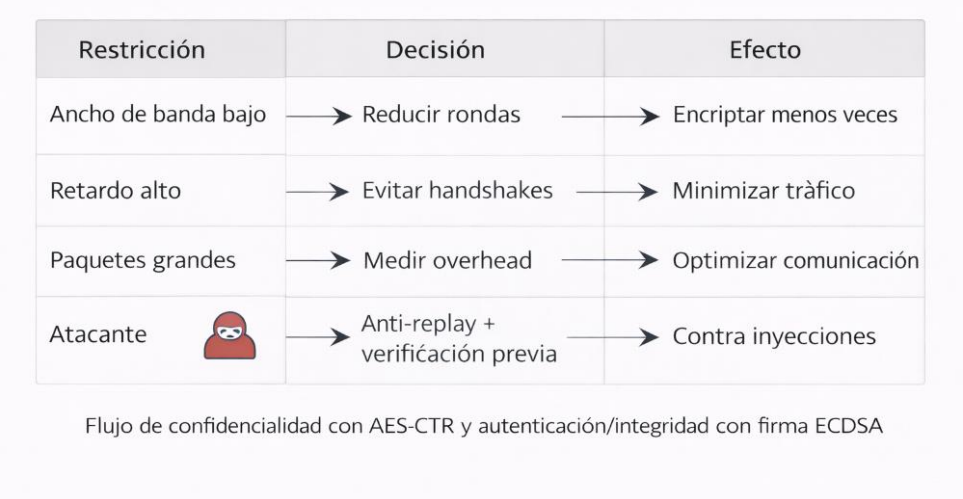
Respecto al tamaño de clave, se adopta AES-128 por coherencia con el nivel de seguridad objetivo asociado a P-256 y por eficiencia computacional. En redes subacuáticas, donde el canal es costoso y los nodos tienen recursos limitados, incrementar el costo criptográfico sin un beneficio proporcional puede degradar latencia y consumo; por ello, la selección prioriza un balance entre fortaleza y desempeño, manteniendo consistencia con recomendaciones de estandarización y perfiles ampliamente implementados (Hankerson et al., 2004; NIST, 2023a).

2.3.5 Consideraciones para adaptar ECIES a UWSN

Adaptar ECIES a redes inalámbricas subacuáticas no implica proponer un cifrador nuevo, sino ajustar su empleo a las restricciones del canal acústico y a las limitaciones de los nodos. En UWSN, el retardo de propagación elevado y la capacidad limitada del medio amplifican el impacto de cualquier incremento en tamaño de paquete o en señalización (handshakes), por lo que el diseño debe priorizar intercambios mínimos y procedimientos verificables bajo condiciones controladas (Akyildiz et al., 2005b; Casari & Zorzi, 2022; Domingo, 2011). La **Figura 3** sintetiza la relación entre estas restricciones del canal acústico y las decisiones adoptadas para la adaptación del esquema ECIES-based (AES-CTR + ECDSA).

Figura 3

Relación entre restricciones del canal acústico y decisiones de adaptación del esquema ECIES-based (CTR + ECDSA)



Una primera decisión es reducir rondas de mensajes: cuanto más “conversación” requiere el protocolo, mayor es el impacto del retardo acústico y mayor la probabilidad de colisiones o pérdidas asociadas a la contención del medio. En segundo lugar, debe controlarse

el tamaño del material criptográfico que acompaña al ciphertext (por ejemplo, el punto público efímero y metadatos), ya que enlaces de baja tasa pueden inducir fragmentación, retransmisiones y aumento del retardo extremo a extremo. Finalmente, al adoptar AES-CTR para confidencialidad y firma ECDSA para autenticación/integridad, el receptor puede aplicar un flujo "verificar → descifrar", descartando temprano mensajes manipulados y evitando consumo innecesario en capas superiores (Hankerson et al., 2004; NIST, 2023a). La **Tabla 8** resume estas consideraciones de adaptación ECIES-based en UWSN y el efecto esperado sobre el overhead y el desempeño del enlace.

Tabla 8

Consideraciones de adaptación ECIES-based en UWSN y efecto esperado

Consideración	**Decisión en la tesis**	**Efecto esperado (métricas)**
Retardo acústico alto	Minimizar handshakes	↓ Delay E2E; ↓ pérdidas por contención
Ancho de banda limitado	Controlar tamaño de cabeceras/cripto-metadatos	↓ Overhead; ↑ Goodput
Riesgo de replay/inyectados	Contador/nonce + firma sobre metadatos	↓ aceptación de tráfico inválido
Nodos restringidos	Verificar firma antes de descifrar	↓ costo de procesar paquetes falsos
Escalabilidad	Evaluar bajo densidad creciente	Medir degradación: Delay, PDR, overhead

Estas decisiones se justifican porque el canal acústico limita la capacidad y eleva el retardo(Akyildiz et al., 2005a; NIST, 2023b) Además, la literatura sobre seguridad en redes

subacuáticas recomienda medir el impacto real de los mecanismos de seguridad en latencia y consumo, no solo su fortaleza criptográfica (Casari & Zorzi, 2022; Domingo, 2011).

2.4. Simulación y evaluación en OMNeT++

El análisis de protocolos criptográficos en redes de sensores inalámbricos subacuáticos (UWSN) enfrenta limitaciones técnicas y logísticas que dificultan su evaluación en entornos reales. Las condiciones hostiles del medio como la variabilidad de la velocidad del sonido, el retardo de propagación, la absorción acústica, el ruido ambiental y el consumo energético hacen que las pruebas físicas sean costosas, complejas y poco reproducibles. Frente a este panorama, la simulación computacional se consolida como una alternativa rigurosa, controlada y eficiente para validar algoritmos de seguridad como ECIES en escenarios UWSN, permitiendo a los investigadores modelar condiciones específicas y evaluar métricas clave como latencia, entrega de paquetes, eficiencia energética y resiliencia frente a ataques (Shin & Park, 2008).

2.4.1 OMNeT++ y su aplicación en contextos subacuáticos

Una de las ventajas de OMNeT++ es su capacidad para implementar módulos personalizados en la capa de aplicación, lo que permite integrar mecanismos de seguridad en el flujo de generación, envío y recepción de mensajes sin alterar el modelo del canal acústico. En esta tesis, esta capacidad se emplea para incorporar un esquema híbrido basado en ECIES, incluyendo el acuerdo de secreto compartido (ECDH), la derivación de claves (KDF) y el cifrado simétrico mediante AES-CTR, complementado con autenticación e integridad a través de firma digital ECDSA (SHA-256) (Barbeau & Garcia-Alfaro, 2015).

A nivel experimental, la modularidad del simulador permite instrumentar métricas relevantes como latencia extremo a extremo, tasa de entrega, sobrecarga en bytes y eventos de aceptación/descartes asociados a la verificación criptográfica. Adicionalmente, cuando se incorpora un modelo de energía o un esquema de estimación, es posible aproximar el costo

energético asociado a transmisión/recepción y al procesamiento criptográfico. Finalmente, el entorno facilita la inclusión de comportamientos atacantes (p. ej., escucha pasiva, inyección de tráfico o suplantación), permitiendo analizar la robustez del mecanismo de seguridad ante amenazas concretas dentro de escenarios controlados.

Se selecciona ECIES porque, al ser un esquema híbrido (ECDH + KDF + cifrado simétrico), permite obtener confidencialidad con overhead controlado, y al incorporar autenticación/integridad se mitigan amenazas típicas (suplantación, manipulación, replay). La simulación en OMNeT++ se utiliza para cuantificar el trade-off seguridad–desempeño bajo condiciones acústicas controladas (retardo, pérdidas, densidad).

2.4.2 Enfoque de evaluación y criterios de análisis

La validación experimental en redes de sensores submarinos (UWSN) enfrenta restricciones operativas importantes: el despliegue físico implica costos elevados, complejidad logística y una alta variabilidad del medio. En este contexto, la simulación se consolida como un enfoque metodológico adecuado para estudiar el desempeño bajo condiciones controladas, permitiendo comparar alternativas y aislar variables de interés sin introducir sesgos por condiciones ambientales no repetibles. Esto es especialmente relevante en UWSN, donde el canal acústico presenta características que degradan el desempeño incluso en ausencia de mecanismos de seguridad: velocidad de propagación baja, ancho de banda limitado, pérdidas dependientes de distancia y frecuencia y comportamiento altamente condicionado por el entorno (Domingo, 2011; Jensen et al., 2011; Urick, 1983).

Desde la perspectiva de seguridad, la evaluación no se limita a describir un esquema criptográfico, sino a cuantificar su impacto sistémico sobre la comunicación. En UWSN, la incorporación de protección extremo a extremo introduce costos observables en al menos tres dimensiones: (i) overhead de transmisión (campos adicionales y crecimiento del tamaño del mensaje), (ii) costo de procesamiento (cifrado/validación), y (iii) efectos indirectos sobre

colas, contención y probabilidad de entrega, particularmente cuando aumenta la densidad de nodos. Por ello, los estudios y marcos de análisis en UWSN suelen enfatizar métricas de red como retardo extremo a extremo, tasa de entrega de paquetes y eficiencia de transmisión, además de métricas asociadas a la sobrecarga del mecanismo de seguridad (Casari & Zorzi, 2022; Domingo, 2011).

En el caso de esquemas híbridos como ECIES, la evaluación en simulación resulta especialmente pertinente porque el esquema combina operaciones de criptografía de curva elíptica con cifrado simétrico y verificación de integridad/autenticación. A nivel conceptual, ECIES se apoya en un acuerdo de secreto compartido (por ejemplo, ECDH), una función de derivación de claves y un cifrado simétrico para confidencialidad, complementado por un mecanismo explícito de autenticación/integridad (Hankerson et al., 2004; International Organization for Standardization & International Electrotechnical Commission, 2006; Standards for Efficient Cryptography Group (SECG), 2009). En una UWSN, estos componentes deben analizarse no solo por su solidez criptográfica, sino por su efecto combinado sobre el tráfico acústico, donde el incremento de carga y la contención pueden amplificar degradaciones en desempeño.

En coherencia con estas consideraciones, esta tesis adopta un enfoque ECIES-based en el cual la confidencialidad del payload se implementa mediante AES en modo CTR, mientras que la integridad y autenticación se garantizan mediante firma digital ECDSA (SHA-256). Este diseño permite aplicar en el receptor un flujo de validación 'verificar -> descifrar', descartando tempranamente mensajes alterados o reinyectados antes de consumir recursos en el descifrado. El análisis se realiza bajo escenarios controlados representativos de UWSN, cuantificando el impacto del esquema en métricas de desempeño y en la sobrecarga asociada a los campos criptográficos transmitidos (Urick, 1983; Jensen et al., 2011; Casari et al., 2022).

2.4.3 Métricas de desempeño relevantes

Para evaluar el impacto de un mecanismo de seguridad en UWSN no basta con describir el algoritmo: es necesario cuantificar cómo cambian las propiedades de la red cuando se habilita protección extremo a extremo. Debido a las restricciones del canal acústico capacidad limitada y latencia elevada, pequeñas variaciones en tamaño de paquete o en temporización pueden amplificarse en el retardo extremo a extremo, la eficiencia de entrega y la utilización efectiva del enlace (Casari & Zorzi, 2022; Domingo, 2011; Urick, 1983). Por ello, este trabajo prioriza métricas que reflejen la relación entre seguridad y desempeño, permitiendo comparar de forma consistente perfiles sin seguridad frente a perfiles ECIES-based bajo condiciones equivalentes. La **Tabla 9** resume las métricas sugeridas para evaluar el perfil ECIES-based (AES-CTR + firma ECDSA) en UWSN y su interpretación a nivel experimental.

Tabla 9

Métricas sugeridas para evaluar ECIES-based (AES-CTR + firma ECDSA) en UWSN

Métrica	**Qué mide**	**Relación con ECIES (AES-CTR + ECDSA)**
Retardo extremo a extremo (E2E delay)	Latencia total desde la generación del mensaje en el emisor hasta la entrega del payload en el receptor.	Puede aumentar por mayor tamaño del mensaje (overhead), contención/colas y por el procesamiento de cifrado/verificación.
Tasa de entrega de paquetes (PDR)	Proporción de paquetes válidos entregados respecto a los transmitidos.	Puede disminuir si el canal se satura por incremento de carga (tamaño/contención) o si existen descartes por

		verificación de integridad en escenarios con ataque.
Overhead (bytes)	Incremento de tamaño del mensaje respecto al perfil sin seguridad.	Incluye punto público efímero, IV/nonce/contador y firma ECDSA (además de metadatos del perfil experimental).
Tiempo de establecimiento de comunicación segura (si aplica)	Tiempo requerido para completar el procedimiento inicial definido por el escenario (p. ej., sincronización/handshake si existe).	Depende del mecanismo inicial modelado en el escenario y del costo adicional asociado a la activación del perfil seguro.
Consumo energético (estimado)	Energía aproximada asociada a transmisión/recepción y al cómputo criptográfico.	Permite discutir viabilidad en nodos restringidos: el overhead incrementa tiempo de radio/acústico activo y el cifrado/verificación añade costo de cómputo.

Nota. Las métricas se alinean con recomendaciones de evaluación práctica en redes subacuáticas y con el análisis de desempeño en entornos acústicos (Casari & Zorzi, 2022; Domingo, 2011; Jensen et al., 2011; Urick, 1983).

Tras definir estas métricas, el análisis experimental se orienta a observar tendencias conforme varían condiciones de carga y densidad. En particular, el overhead criptográfico no debe evaluarse aisladamente: su efecto se manifiesta en la red cuando la contención aumenta y el canal se aproxima a su capacidad efectiva. En UWSN, estas condiciones suelen provocar

incrementos no lineales en el retardo y caídas en entrega, por lo que resulta fundamental reportar los resultados de manera comparativa y bajo configuraciones equivalentes. (Casari & Zorzi, 2022; Domingo, 2011).

Finalmente, cuando se incorpora un escenario con atacante, la métrica “tasa de descartes por validación” permite distinguir pérdidas por condiciones del canal de pérdidas inducidas por tráfico inválido (inyección o replay). Esta separación es relevante porque en la propuesta adoptada el receptor realiza verificación criptográfica previa al descifrado; por tanto, el rechazo por firma inválida se convierte en un evento observable que afecta directamente a la entrega efectiva y al retardo (por congestión adicional o por intentos repetidos de transmisión, según el comportamiento del escenario). (Dini & Lo Duca, 2012).

Capítulo III: Desarrollo de la Propuesta

Este capítulo presenta el diseño metodológico, la ingeniería de requerimientos, la arquitectura de seguridad y el diseño experimental empleados para integrar y evaluar un esquema ECIES-based en una red inalámbrica de sensores subacuáticos (UWSN) simulada en OMNeT++. En lugar de describir procedimientos operativos paso a paso, se organiza el contenido en torno a decisiones de diseño, supuestos, criterios verificables y condiciones de reproducibilidad, de modo que el lector pueda comprender qué se propuso, por qué se seleccionó dicho enfoque y en qué condiciones se evaluó su impacto en el desempeño de la red.

3.1. Metodología del desarrollo

El desarrollo de la propuesta se orienta a implementar y evaluar un esquema de seguridad basado en ECIES dentro de un entorno de simulación de redes inalámbricas subacuáticas (UWSN), utilizando OMNeT++ y sus extensiones para modelado de red y canal acústico. En este contexto, la metodología se entiende como el conjunto organizado de procedimientos, técnicas y herramientas que permiten ejecutar el desarrollo con un orden definido, asegurando control del proceso y coherencia entre lo diseñado, lo implementado y lo validado.

Dado que la propuesta combina componentes de distinta naturaleza (criptografía, arquitectura de red, integración en simulador y experimentación), se adoptó un enfoque iterativo e incremental, en el que la solución se construye por bloques verificables: desde escenarios mínimos de conectividad hasta topologías más representativas, incorporando progresivamente el cifrado, la autenticación y la medición de desempeño. La **Figura 4**

resume el flujo de trabajo adoptado bajo la metodología Kanban y la organización incremental de las tareas de implementación y validación. La evaluación experimental se ejecutó en OMNeT++ 6.1, organizando el entorno en un modelo base y una capa de evaluación para asegurar comparabilidad entre perfiles.

Figura 4

Desarrollo según la metodología Kanban

Para organizar, controlar y documentar este avance incremental se adopta un enfoque ágil basado en Kanban, el cual permite visualizar el estado del trabajo, priorizar actividades y sostener ciclos de mejora continua durante el desarrollo.

3.1.1 Gestión del desarrollo mediante Kanban

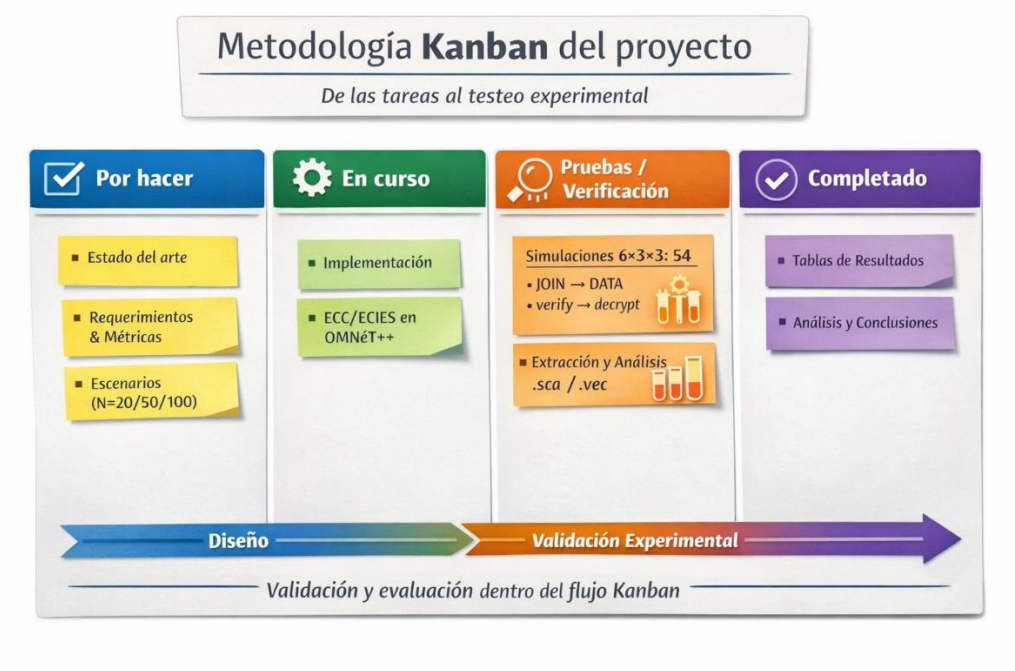
El control del proyecto se realiza a través de un tablero Kanban, donde cada actividad se representa como una tarjeta (task card) que avanza según su estado. Para este trabajo se proponen columnas que reflejen el ciclo real de construcción y evaluación del sistema. La **Figura 5** ilustra la estructura del tablero Kanban propuesto y la progresión de tareas utilizada durante la implementación y validación incremental del sistema:

- Backlog técnico: requerimientos del sistema, decisiones criptográficas, tareas de instrumentación, y cambios sobre módulos de aplicación.
- En progreso: tareas activas con WIP limitado.
- Verificación: pruebas de compilación, ejecución, coherencia de formato de mensajes, y validación mínima del flujo “verificar → descifrar”.

- Listo / Hecho: tareas cerradas con evidencia (configuración guardada, resultados mínimos exportables, notas de decisión).

Figura 5

Desarrollo según la metodología Kanban



Las tarjetas se redactan con un enfoque orientado a resultados, especificando: objetivo de la tarea, entradas necesarias, criterio de aceptación y artefacto de salida (por ejemplo: archivo .ned, configuración .ini, clase C++ del módulo criptográfico, scripts de extracción de métricas o tabla de resultados). De esta manera, Kanban no solo gestiona el avance, sino que también facilita la trazabilidad de lo construido y su evidencia experimental. La **Tabla 10** resume la estructura recomendada para las tarjetas Kanban y los campos mínimos utilizados para documentar tareas y evidencias del desarrollo.

Tabla 10

Desarrollo según la metodología Kanban

Etapa	**Descripción**	**Criterio de completitud (DoD)**
Por hacer	Tareas definidas y priorizadas para el incremento.	Actividad descrita y aceptada para ejecución.
En progreso	Implementación y ajustes del módulo o configuración.	Compila y ejecuta sin errores; evidencia mínima.
En verificación	Pruebas de funcionamiento bajo escenario controlado.	Resultados coherentes con el comportamiento esperado.
Hecho	Actividad finalizada y documentada.	Evidencia en texto/figuras/tablas y configuración reproducible.

3.1.2 Validación incremental del sistema

La validación incremental se aplicó como un proceso de aseguramiento de funcionalidad y consistencia interna del sistema, con el objetivo de evitar que los resultados de desempeño (Capítulo 4) se vean afectados por errores de integración o por inconsistencias en el formato de mensajes. En términos prácticos, la validación se estructuró en tres niveles, cada uno con criterios de aceptación verificables y trazables a la configuración de simulación.

3.1.2.1 Nivel I: Verificación funcional del escenario base (sin seguridad).

Se estableció un escenario de referencia donde la aplicación transmite y recibe mensajes sin incorporar mecanismos criptográficos. Este nivel valida: (i) conectividad entre nodos (topología, direccionamiento y rutas si aplica), (ii) compatibilidad del stack de comunicación con el modelo de canal acústico configurado, y (iii) integridad estructural del mensaje de aplicación (campos, tamaños y secuencia de envío/recepción). El propósito es confirmar que cualquier variación posterior en métricas de red provenga del mecanismo de seguridad o del factor experimental, y no de fallos iniciales del sistema.

3.1.2.2 Nivel II: Integración del bloque criptográfico y verificación del flujo operativo "verificar → descifrar".

En este nivel se integra el esquema ECIES-based (ECDH + KDF + AES-CTR + firma ECDSA) como lógica de la capa de aplicación. La validación se centra en la corrección del pipeline criptográfico y en la preservación de invariantes operacionales:

- El emisor debe: derivar secreto (ECDH), derivar clave simétrica (KDF), cifrar payload (AES-CTR) y generar firma (ECDSA) sobre el *ciphertext* y metadatos críticos.
- El receptor debe ejecutar la política fail-fast: verificar firma ECDSA antes de descifrar; si la firma falla, el mensaje se descarta sin aplicar AES-CTR ni derivación de claves posterior al descarte.

  Este criterio es esencial para asegurar coherencia con el modelo de amenazas y para habilitar métricas operativas de seguridad (p. ej., tasa de mensajes rechazados por autenticidad). Además, se controla explícitamente la ambigüedad terminológica: cualquier mención a "MAC" en el stack de red se interpreta como *Medium Access Control* (capa de enlace) y no como *Message Authentication Code*.

3.1.2.3 Nivel III: Escalamiento controlado y validación bajo complejidad creciente (densidad y atacante).

Una vez comprobada la corrección del flujo seguro, se incrementa la complejidad del entorno manteniendo comparabilidad: (i) aumento de densidad de nodos y/o carga de tráfico, y (ii) incorporación de perfiles con atacante (escucha, manipulación, reinyección). El objetivo de este nivel no es "probar invulnerabilidad", sino verificar que el sistema mantiene un comportamiento consistente frente a perturbaciones: los mensajes manipulados o falsificados deben ser descartados por verificación de firma, y el sistema debe continuar operando bajo condiciones acústicas restringidas sin romper el formato ni el control de ejecución.

3.1.3 Estrategia de validación incremental

La construcción de la propuesta se valida por iteraciones, evitando integrar todo el sistema "de una sola vez". El principio es asegurar primero la conectividad y el funcionamiento básico del escenario, y luego incorporar progresivamente el costo criptográfico y la escalabilidad de la topología. A nivel metodológico, la validación sigue una progresión típica:

- **Iteración 1 (conectividad mínima):** escenario base con pocos nodos para verificar comunicación y configuración del canal.
- **Iteración 2 (seguridad punto a punto):** incorporación del módulo ECIES en un enlace simple para validar cifrado/descifrado correcto y consistencia de claves derivadas.
- **Iteración 3 (topología estrella con boya):** integración del esquema de seguridad en una arquitectura con nodo central y múltiples sensores, evaluando concurrencia de tráfico.
- **Iteración 4 (escalabilidad):** aumento gradual del número de nodos y carga de la red para observar efectos de colisiones, latencia y tasa de éxito bajo mayores densidades.

En cada iteración se establecen criterios de aceptación (por ejemplo: porcentaje mínimo de descifrado exitoso, consistencia de autenticación, latencia máxima tolerable o estabilidad del escenario) y se documentan los resultados para alimentar las secciones de análisis posteriores.

3.1.4 Gestión de cambios e incidencias

Como resultado de la metodología aplicada, se generan y conservan evidencias técnicas que respaldan la reproducibilidad del trabajo, entre ellas:

- Artefactos de diseño (diagramas de arquitectura, flujo criptográfico, definición de parámetros).
- Implementaciones y configuraciones del simulador (.cc/.h, .ned, .ini).
- Registros de ejecución y métricas (vectores, scalars, exportación a CSV/Excel).
- Evidencias del tablero Kanban (estado de tareas, trazabilidad de entregables y control de iteraciones).

Con ello, el capítulo no solo describe "qué se implementó", sino también cómo se gestionó y validó el proceso de adaptación del mecanismo criptográfico a las restricciones propias de una red subacuática simulada.

3.2 Ingeniería y Análisis de Requerimientos del Sistema

La adaptación de un esquema criptográfico basado en curvas elípticas para redes inalámbricas subacuáticas requiere establecer, antes del diseño e implementación, un conjunto de requerimientos verificables que delimiten el alcance técnico, las restricciones del entorno y los criterios de validación. En redes UWSN, el canal subacuático se caracteriza por alta latencia, ruido, variabilidad y limitaciones de ancho de banda; por ello, los mecanismos de seguridad deben mantener fortaleza criptográfica sin introducir una sobrecarga inviable para nodos con recursos restringidos (Akyildiz et al., 2005b).

En esta sección se documenta la ingeniería de requerimientos siguiendo una clasificación por niveles (stakeholders, sistema y plataforma), con el objetivo de: (a) asegurar trazabilidad entre necesidades, funciones y herramientas; (b) facilitar la gestión de cambios durante el desarrollo; y (c) establecer criterios de verificación y validación medibles mediante simulación. El marco de organización de requerimientos se alinea con la lógica de nomenclatura y trazabilidad propuesta en ISO/IEC/IEEE 29148:2018 para ingeniería de requerimientos(ISO/IEC/IEEE 29148-2018: Systems and Software Engineering — Life Cycle Processes — Requirements Engineering, 2018).

3.2.1 Abreviaturas y nomenclatura de requerimientos (StSR, SySR, SRSH)

Para estructurar los requerimientos y mantener trazabilidad entre niveles, se adopta la nomenclatura mostrada en la **Anexo A. Ficha de requerimientos**, diferenciando: requerimientos de stakeholders (necesidades), requerimientos del sistema (qué debe hacer la solución) y requerimientos de hardware/software (con qué se implementa y ejecuta). Esta separación permite que cada requerimiento sea único, verificable y rastreable hacia su origen y su implementación. Para ello se implemento e investigo sobre (ISO/IEC/IEEE 29148-2018: Systems and Software Engineering — Life Cycle Processes — Requirements Engineering, 2018) el cual complemento los requerimientos necesarios detallados en el **Anexo A. Ficha de requerimientos.** A continuación se puede tomar un resumen en el cual se especifica como se va a trabajar de este planteamiento y clasificación.

Abreviaturas y clasificación de requerimientos

Abreviatura	Descripción
StSR	Requerimientos de Stakeholders
SySR	Requerimientos del Sistema
SRSH	Requerimientos de Hardware y Software

Formato de identificación (ID):

- StSR-XX: requerimientos de stakeholders (ej.: StSR-01, StSR-02).
- SySR-XX: requerimientos del sistema (ej.: SySR-01, SySR-02).
- SRSH-XX: requerimientos de plataforma/herramientas (ej.: SRSH-01, SRSH-02).

Atributos mínimos por requerimiento (plantilla recomendada):

- ID, enunciado ("El sistema deberá…"), tipo (funcional / no funcional / restricción), prioridad (Alta/Media/Baja), criterio de verificación, trazabilidad (ID origen/destino) y observaciones.

3.2.2 Requerimientos de Stakeholders (StSR)

Los requerimientos de stakeholders expresan las necesidades del proyecto desde el punto de vista académico, técnico y de investigación aplicada también definido en la **Tabla 11**, considerando que el sistema será validado mediante simulaciones UWSN y no mediante despliegues físicos. En esta tesis, los stakeholders principales son:

- **Autor/investigador** (desarrollo e integración del módulo criptográfico). El cargo de apartado es el Sr. Vásconez Andrés.
- **Director académico** (validación metodológica, técnica y científica). El cargo de este apartado con dirección académica es el MSc. Cuzme Fabián.
- **Asesor académico** (evaluación de coherencia, trazabilidad y reproducibilidad). El cargo de este apartado con asesoría académica es el MSc. Suárez Luis.

Tabla 11

Requerimientos funcionales de stakeholders (StSR)

ID	Requerimiento (Stakeholders)	Prioridad
StSR-01	El proyecto deberá permitir proteger datos transmitidos en una red UWSN mediante un esquema criptográfico basado en ECC/ECIES.	Alta
StSR-02	El proyecto deberá permitir evaluar el impacto del cifrado en el desempeño de la red (latencia, entrega, velocidad de transmisión y procesamiento).	Alta
StSR-03	El sistema deberá integrarse como un módulo funcional (Escenario de simulación) dentro de un entorno de simulación de redes (OMNeT++).	Media
StSR-04	El proyecto deberá permitir comparación entre un escenario base (sin cifrado) y uno seguro (con ECIES).	Alta
StSR-05	El proyecto deberá contemplar una arquitectura con nodos sensores y una boya/sink como receptor central.	Alta
StSR-06	El proyecto deberá permitir escalabilidad del número de nodos para analizar comportamiento bajo diferentes densidades.	Media
StSR-07	El proyecto deberá generar resultados reproducibles (parámetros, escenarios y semillas documentadas).	Alta

Nota: La orientación a simulación se justifica porque las pruebas físicas en UWSN son costosas y poco reproducibles; la simulación permite controlar variables del canal y medir métricas relevantes de forma (Shin & Park, 2008; Xie et al., 2021b).

3.2.2.1 Requerimientos funcionales (StSR)

Además de los requerimientos funcionales, el sistema debe satisfacer un conjunto de requerimientos no funcionales definidos por los stakeholders, los cuales establecen restricciones y criterios de calidad asociados a desempeño, seguridad, robustez, usabilidad experimental y trazabilidad. Estos requerimientos guían tanto la implementación como la evaluación, ya que delimitan condiciones mínimas de operación y sirven como base para justificar decisiones de diseño (p. ej., control de overhead, verificabilidad del flujo verify → decrypt y repetibilidad de experimentos). La **Tabla 12** presenta los requerimientos no funcionales de stakeholders (StSR) considerados en este trabajo.

Tabla 12

Requerimientos no funcionales de stakeholders (StSR)

ID	Requerimiento (Stakeholders)	Prioridad
StSR-08	La solución propuesta deberá privilegiar eficiencia en entornos con recursos limitados, justificando el uso de ECC por tamaño de claves y costo computacional.	Media
StSR-09	La implementación deberá ser modular y extensible para permitir futuras variantes (curvas alternativas, ataques simulados, movilidad).	Baja
StSR-10	El desarrollo deberá mantener claridad metodológica (toma de requerimientos, trazabilidad y criterios de validación).	Alta
StSR-11	El sistema deberá representar un canal subacuático con condiciones que afecten la comunicación (atenuación, retardo, ruido) para analizar el impacto real del cifrado.	Alta

3.2.2.2 Requerimientos no funcionales (StSR)

El planteamiento experimental y la implementación del esquema ECIES-based se desarrollan bajo un conjunto explícito de restricciones y supuestos definidos por los stakeholders. Estas condiciones delimitan el alcance del trabajo y establecen el marco bajo el cual los resultados deben interpretarse, incluyendo aspectos como las limitaciones del canal acústico simulado, el modelo de amenazas asumido (sin compromiso de claves privadas), y las decisiones de configuración necesarias para asegurar comparabilidad entre perfiles (NS vs S). La **Tabla 13** resume las restricciones y supuestos (StSR) adoptados para el desarrollo y la evaluación.

Tabla 13

Restricciones y supuestos (StSR)

ID	Restricción / supuesto	Prioridad
StSR-12	La validación se realizará mediante simulación y no mediante hardware real (alcance del trabajo).	Media
StSR-13	Se asume un esquema con receptor central (boya/sink) y nodos sensores que envían datos hacia él (topología base).	Alta
StSR-14	Se asume que las claves públicas necesarias para ECIES están disponibles para los nodos según el escenario (preconfiguración en simulación).	Media
StSR-15	Se prioriza un esquema híbrido (ECC) confidencialidad con AES-CTR + integridad/autenticación con firma ECDSA (Velázquez et al., 2021; Zhang et al., 2022).	Alta

3.2.3 Requerimientos del Sistema (SySR)

Los requerimientos del sistema traducen las necesidades de stakeholders a especificaciones técnicas implementables y comprobables. En este proyecto, el sistema corresponde al módulo criptográfico ECIES integrado a nodos UWSN dentro del entorno OMNeT++/INET, operando principalmente a nivel de aplicación para cifrar y autenticar mensajes antes de su transmisión acústica.

3.2.3.1 Requerimientos funcionales (SySR)

Los requerimientos funcionales del sistema describen qué debe hacer el módulo ECIES-based dentro del nodo UWSN y cómo debe comportarse durante el ciclo operativo de comunicación (registro, transmisión y recepción). En particular, estos requerimientos especifican funciones verificables como: encapsular mensajes bajo el perfil seguro, incorporar el material criptográfico necesario (p. ej., IV/nonce, clave pública efímera y firma), ejecutar validación previa (*verify* → *decrypt*), y garantizar la entrega del payload únicamente cuando el mensaje sea auténtico e íntegro. Asimismo, los requerimientos funcionales definen puntos de instrumentación para registrar eventos y facilitar la evaluación experimental dentro de OMNeT++/INET. La **Tabla 14** presenta los requerimientos funcionales del sistema (SySR) considerados para la implementación y validación.

Tabla 14

Requerimientos funcionales del sistema (SySR)

ID	Requerimiento del sistema	Prioridad
SySR-01	El sistema deberá implementar ECIES para cifrado híbrido, incluyendo componente ECC y un esquema simétrico para confidencialidad (AES-CTR) e integridad/autenticación mediante firma digital ECDSA (SHA-256).	Alta

SySR-02	El sistema deberá generar una clave efímera por sesión o por mensaje, y calcular el secreto compartido mediante ECIES.	Alta
SySR-03	El sistema deberá derivar claves simétricas a partir del secreto compartido usando una función hash/KDF (p. ej., SHA-256).	Alta
SySR-04	El sistema deberá cifrar la carga útil usando AES en modo CTR y generar una firma digital ECDSA (SHA-256) sobre el ciphertext y metadatos relevantes (por ejemplo, IV/nonce y contador/identificador anti-replay cuando aplique).	Media
SySR-05	El sistema deberá encapsular y transmitir un mensaje seguro con al menos: R (punto público efímero), IV/nonce/contador, C (ciphertext) y Sig (firma ECDSA).	Alta
SySR-06	El receptor (boya/sink) deberá reconstruir el secreto, verificar la firma ECDSA y descifrar; si la verificación falla, deberá descartar el paquete.	Alta
SySR-07	El módulo criptográfico deberá integrarse a la capa de aplicación del modelo OMNeT++ para cifrar antes de transmitir y descifrar al recibir.	Baja
SySR-08	El sistema deberá registrar métricas por evento: tiempo de cifrado/descifrado, éxito de verificación y latencia total por paquete.	Alta

SySR-09	El sistema deberá permitir habilitar/deshabilitar el módulo de seguridad para obtener escenarios sin seguridad vs seguro.	Media
SySR-10	El sistema deberá soportar múltiples tamaños de red (escenarios escalables) definidos en archivos de configuración.	Baja

3.2.3.2 Requerimientos no funcionales (SySR)

Los requerimientos no funcionales del sistema establecen atributos de calidad y restricciones técnicas que deben cumplirse durante la integración del esquema ECIES-based en el entorno OMNeT++/INET. Estos requerimientos no describen funciones específicas, sino condiciones como eficiencia computacional, control del overhead (tamaño adicional por encapsulado), consistencia del flujo *verify* → *decrypt*, reproducibilidad de ejecuciones y trazabilidad de resultados para análisis posterior. En el contexto de UWSN, estas propiedades son especialmente relevantes debido a la latencia elevada y la capacidad limitada del canal acústico, donde incrementos moderados de carga pueden afectar el desempeño global del escenario. La **Tabla 15** resume los requerimientos no funcionales del sistema (SySR) definidos para guiar la implementación y la evaluación.

Tabla 15

Requerimientos no funcionales del sistema (SySR)

ID	Requerimiento no funcional	Prioridad
SySR-11	El sistema deberá mantener una sobrecarga de seguridad medible y reportable, para evaluar viabilidad en UWSN	Alta

SySR-12	El módulo criptográfico deberá ser modular (bajo acoplamiento), permitiendo sustitución de curva o modo de cifrado sin rediseñar toda la red.	Media
SySR-13	El sistema deberá ser robusto ante pérdida/colisiones: fallos de recepción o verificación no deben detener la simulación.	Alta
SySR-14	El modelo deberá permitir reproducibilidad experimental (parámetros documentados, escenarios repetibles).	Media

3.2.3.3 Requerimientos de interfaces y comunicación (SySR)

Los requerimientos de interfaces y comunicación especifican cómo intercambian información los módulos del sistema dentro del entorno OMNeT++/INET, definiendo el formato de mensajes, los campos mínimos obligatorios y las reglas de interacción entre aplicaciones (FullApp/Gwapp) y la pila de comunicación subyacente. En el contexto ECIES-based, estos requerimientos aseguran que el encapsulado criptográfico sea interoperable, que los mensajes conserven su semántica (JOIN, JOIN_REPLY y DATA) y que el receptor pueda aplicar de forma determinista la política verify → decrypt antes de aceptar y entregar el payload. Asimismo, delimitan los elementos que deben registrarse para instrumentación y análisis (p. ej., tamaños de payload, eventos de verificación, descartes y tiempos de procesamiento). La **Tabla 16** presenta los requerimientos de interfaces y comunicación (SySR) considerados en la integración del sistema.

Tabla 16

Requerimientos de interfaces y comunicación (SySR)

ID	Requerimiento de interfaz	Prioridad
SySR-15	El sistema deberá definir el formato del mensaje seguro y su serialización (R, C, Sig y metadatos como IV/nonce y nonce/contador/seq si aplica).	Alta
SySR-16	El módulo deberá interoperar con el stack de red del entorno (INET/UAN) sin romper el flujo normal de envío/recepción.	Alta
SySR-17	El sistema deberá permitir configurar por parámetros: longitud de claves, tamaño de payload, tasa de envío y número de nodos.	Baja
SySR-18	El receptor deberá disponer de un mecanismo para acceder a su clave privada y a la información necesaria para descifrar de forma consistente.	Media

3.2.3.4 Requerimientos de seguridad (SySR)

Los requerimientos de seguridad se establecen considerando amenazas típicas en UWSN (intercepción, manipulación, suplantación y ataques activos sobre enlaces con alto retardo), y se alinean a objetivos de confidencialidad, integridad y autenticación propuestos en la literatura de seguridad subacuática (Dini & Lo Duca, 2012; Velázquez et al., 2021; Wahid et al., 2021; Zhang et al., 2022). En particular, estos requerimientos delimitan qué información debe permanecer protegida, en qué condiciones un mensaje debe aceptarse o descartarse, y qué evidencia debe registrarse para validar el cumplimiento del flujo verify → decrypt durante la ejecución experimental. La **Tabla 17** resume los requerimientos de seguridad del sistema (SySR) definidos para la implementación del perfil ECIES-based.

Tabla 17

Requerimientos de seguridad del sistema (SySR)

ID	Requerimiento de seguridad	Prioridad
SySR-19	El sistema deberá garantizar confidencialidad del payload transmitido.	Alta
SySR-20	El sistema deberá garantizar integridad y autenticación de mensajes mediante etiqueta criptográfica verificable.	Alta
SySR-21	El sistema deberá mitigar replay mediante nonces/contadores o mecanismo equivalente (en simulación).	Media
SySR-22	El esquema deberá proveer secreto hacia adelante (forward secrecy) usando componentes efímeros en ECIES.	Media
SySR-23	El sistema deberá permitir modelar o discutir resistencia frente a eavesdropping y manipulación en el enlace acústico (amenazas base).	Alta

3.2.4 Requerimientos de Hardware y Software (SRSH)

Dado que la validación se realiza por simulación, los requerimientos SRSH se enfocan en: (a) herramientas de simulación, (b) librerías y entorno de desarrollo, y (c) recursos mínimos de cómputo para ejecutar escenarios con múltiples nodos.

3.2.4.1 Requerimientos de software (SRSH)

Los requerimientos de software describen las condiciones técnicas necesarias para implementar, compilar y ejecutar la solución dentro del entorno de simulación, incluyendo dependencias, compatibilidad de versiones y componentes mínimos del *toolchain*. En este trabajo, tales requerimientos aseguran que el módulo criptográfico ECIES-based se integre de forma consistente con OMNeT++/INET, que los escenarios puedan reproducirse mediante

configuraciones *.ini* y que los artefactos de salida (resultados, registros y métricas) puedan generarse y procesarse para el análisis experimental. La **Tabla 18** presenta los requerimientos de software (SRSH) considerados en el desarrollo y ejecución de la simulación.

Tabla 18

Requerimientos de software (SRSH)

ID	Requerimiento de software	Prioridad
SRSH-01	Se deberá disponer de un entorno de simulación OMNeT++ para modelado de red.	Alta
SRSH-02	Se deberá contar con INET Framework para soporte de capas y modelos de red.	Media
SRSH-03	Se deberá disponer de un modelo/extensión tipo UAN (Underwater Acoustic Network) para enlaces acústicos.	Alta
SRSH-04	Se deberá disponer de un entorno C++ (compilador y depurador) para implementar el módulo criptográfico.	Alta
SRSH-05	Se deberá disponer de herramientas para exportar y procesar métricas (IDE OMNeT++ / CSV / hojas de cálculo).	Media

3.2.4.2 Requerimientos del entorno de simulación (SRSH)

Los requerimientos del entorno de simulación especifican las condiciones mínimas de infraestructura y configuración necesarias para ejecutar los escenarios UWSN de forma consistente, incluyendo el *workspace* de OMNeT++, librerías requeridas, parámetros de compilación, y capacidades de instrumentación para recolectar resultados (p. ej., archivos de salida, trazas/eventlog y métricas). En el contexto de este trabajo, estos requerimientos garantizan que los perfiles NS y ECIES-based se ejecuten bajo un marco homogéneo de canal

y pila de comunicación, permitiendo comparaciones válidas entre ejecuciones. La **Tabla 19** resume los requerimientos del entorno de simulación (SRSH) definidos para la implementación y evaluación del sistema.

Tabla 19

Requerimientos del entorno de simulación (SRSH)

ID	Requerimiento del entorno	Prioridad
SRSH-06	El entorno deberá permitir configurar parámetros del canal acústico: retardo de propagación, atenuación y ruido, para análisis realista.	Alta
SRSH-07	El entorno deberá permitir ejecución de múltiples escenarios (variación de nodos/tráfico) y recolección de métricas por corrida.	Media
SRSH-08	El entorno deberá permitir integrar módulos personalizados a nivel de aplicación para instrumentación de métricas criptográficas.	Alta

3.2.4.3 Requerimientos de hardware/plataforma de ejecución (SRSH)

Los requerimientos de hardware y plataforma de ejecución definen las condiciones mínimas del equipo donde se compila y ejecuta OMNeT++/INET y se procesan los resultados de simulación. En este trabajo, estas especificaciones aseguran que las ejecuciones puedan completarse dentro de tiempos razonables, que la carga de escenarios con mayor densidad no degrade la estabilidad del entorno, y que exista capacidad suficiente para almacenar y analizar artefactos generados (archivos de resultados, registros y trazas). La **Tabla 20** resume los requerimientos de hardware/plataforma (SRSH) establecidos para soportar la ejecución reproducible de los escenarios.

Tabla 20

Requerimientos de hardware/plataforma (SRSH)

ID	Requerimiento de hardware	Prioridad
SRSH-09	El equipo de ejecución deberá contar con CPU multi-núcleo (≥ 4 núcleos) para ejecutar escenarios con decenas de nodos en tiempo razonable.	Media
SRSH-10	El equipo deberá contar con memoria RAM suficiente (≥ 8 GB) para compilar y simular escenarios UWSN con INET/UAN.	Baja
SRSH-11	Se deberá contar con almacenamiento suficiente para resultados y trazas (archivos de salida y vectores).	Bajaa

3.2.5 Matriz de trazabilidad (StSR ↔ SySR ↔ SRSH)

La matriz de trazabilidad permite verificar que cada necesidad de stakeholders (StSR) se implementa mediante requerimientos del sistema (SySR) y se soporta por requerimientos de plataforma (SRSH). Esta trazabilidad es clave para gestión de alcance y para justificar decisiones de implementación y simulación. (ISO/IEC/IEEE 29148-2018: Systems and software engineering — Life cycle processes — Requirements engineering, 2018). La **Tabla 21** presenta un extracto de la matriz de trazabilidad utilizada, evidenciando la correspondencia StSR ↔ SySR ↔ SRSH y los vínculos mínimos necesarios para validación.

Tabla 21

Matriz de trazabilidad (extracto)

StSR	SySR relacionados	SRSH relacionados
StSR-01	SySR-01, SySR-02, SySR-03, SySR-04, SySR-05, SySR-06	SRSH-01, SRSH-04
StSR-02	SySR-08, SySR-10, SySR-11	SRSH-05, SRSH-07
StSR-03	SySR-07, SySR-16	SRSH-01, SRSH-02
StSR-04	SySR-09, SySR-10	SRSH-07
StSR-11	SySR-13, SySR-16	SRSH-03, SRSH-06

3.2.6 Criterios de verificación y validación de requerimientos

Para asegurar que los requerimientos definidos son comprobables, se adopta un enfoque de verificación y validación (V&V) basado en: inspección, análisis, pruebas en simulación y demostración mediante escenarios controlados.

Criterios propuestos:

1. Inspección documental (ID): Revisión de consistencia, claridad y no ambigüedad del enunciado del requerimiento, así como su trazabilidad a requerimientos superiores.
2. Análisis (AN): Evaluación teórica de coherencia con restricciones UWSN (retardo, ruido, ancho de banda) y con seguridad requerida en enlaces acústicos (Akyildiz et al., 2005b; Dini & Lo Duca, 2012).
3. Prueba en simulación (PS): Ejecución de escenarios para comprobar funcionalidad (cifrado/descifrado correcto), robustez (fallo controlado ante firma inválida) y recolección de métricas (tiempo de cifrado/descifrado, latencia, tasa de entrega).
4. Demostración (DM): Evidencia de integración en OMNeT++/INET/UAN (mensajes cifrados en emisor y descifrados en receptor dentro del flujo normal de simulación).

Se puede obtener un resumen mas detallada en la **Tabla 22**.

Tabla 22

Métodos de V&V por tipo de requerimiento (guía)

Tipo de requerimiento	**Método principal**	**Evidencia esperada**
StSR	ID + AN	Matriz de trazabilidad, justificación teórica
SySR funcional	PS + DM	Escenarios controlados y logs de cifrado/descifrado
SySR seguridad	PS + AN	Pruebas de integridad (firma ECDSA), replay si aplica, análisis de amenazas
SRSH	ID + DM	Versiones, instalación, configuración y reproducibilidad

3.3 Diseño de la Arquitectura de Seguridad

Con base en los requerimientos definidos en la Sección 3.2 Ingeniería y Análisis de Requerimientos del Sistema y en la metodología de desarrollo descrita en la Sección 3.1. Metodología del desarrollo, en este apartado se presenta el diseño de la arquitectura de seguridad propuesta para la red inalámbrica subacuática. El objetivo principal de esta arquitectura es garantizar la confidencialidad, integridad y autenticidad de los datos intercambiados entre nodos sensores y el nodo central (boya), manteniendo un impacto controlado sobre el desempeño de la red acústica.

La arquitectura se concibe de forma modular y escalable, de modo que el esquema de seguridad pueda integrarse de manera transparente sobre la arquitectura de comunicación subacuática, sin modificar el comportamiento básico del canal ni del modelo de red, y permitiendo su evaluación en escenarios con diferente densidad de nodos.

3.3.1 Topología de la Red Subacuática

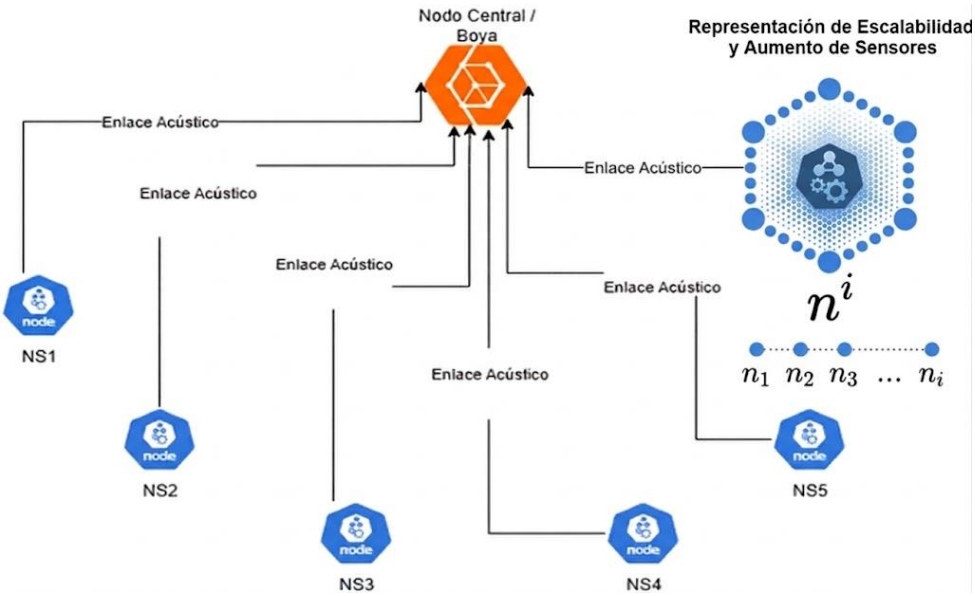
La arquitectura de comunicación adoptada para la propuesta corresponde a una topología radial (estrella), en la cual existe un nodo central (boya) que coordina la red y múltiples nodos sensores (NS) distribuidos en el área de interés. Cada nodo sensor establece un enlace acústico directo con la boya, evitando la necesidad de múltiples saltos y, con ello, reduciendo complejidad de enrutamiento y señalización, aspecto particularmente relevante cuando el canal presenta retardo elevado y disponibilidad limitada de tasa de datos (Chitre et al., 2008; Stojanovic & Chitre, 2018). Esta elección reduce complejidad de ruteo y señalización, y es consistente con la necesidad de minimizar intercambios en un canal con retardo elevado. La densidad se parametrizó para evaluar escalabilidad hasta 100 nodos.

En la

Figura 6 se presenta un esquema representativo de la topología estrella (NS1–NS6) utilizada como referencia conceptual. No obstante, para efectos de evaluación y escalabilidad, el diseño considera un número variable de nodos sensores (N), incrementándose progresivamente hasta alcanzar escenarios de alta densidad (p. ej., N=100) en la fase de simulación, manteniendo el mismo principio de coordinación centralizada por parte de la boya.

Figura 6

Arquitectura general de comunicación acústica con nodo central (Boya) y N^i *nodos sensores.*



Desde la perspectiva funcional, la boya actúa como sumidero (sink/gateway), recolectando la información generada por los sensores y habilitando, según el caso de uso, tareas adicionales como agregación de datos, gestión de sincronización, o interfaz hacia un sistema de superficie. Los nodos sensores, por su parte, se asumen como dispositivos con capacidades restringidas, tanto en energía como en cómputo, motivo por el cual el mecanismo de seguridad debe minimizar rondas de mensajes y tamaño de sobrecarga criptográfica (Goyal et al., 2022; Ranjan et al., 2022).

Las

Figura 7, 8 y 9 presentan el diagrama de flujo del proceso criptográfico ECIES-based implementado en OMNeT++, detallando las etapas de acuerdo de secreto (ECDH), derivación de claves (KDF), cifrado con AES-CTR y autenticación mediante firma/verificación ECDSA.

Figura 7

Diagrama de flujo del proceso criptográfico ECIES-based en OMNeT++: ECDH, derivación KDF, cifrado AES-CTR y verificación/firma ECDSA parte A, Nodo.

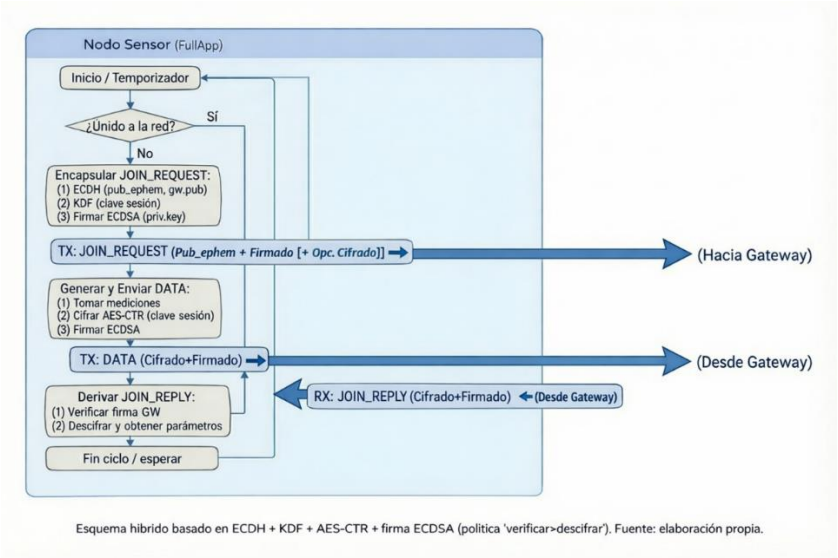


Figura 8

Diagrama de flujo del proceso criptográfico ECIES-based en OMNeT++: ECDH, derivación KDF, cifrado AES-CTR y verificación/firma ECDSA parte B, Gateway.

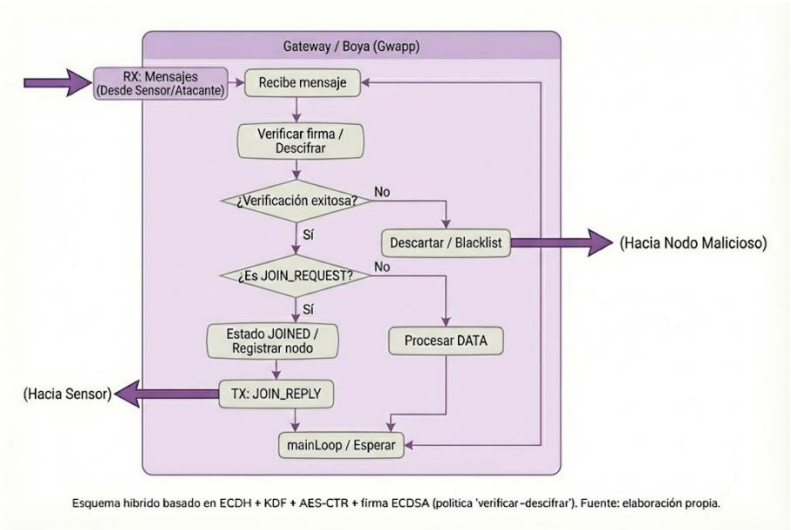
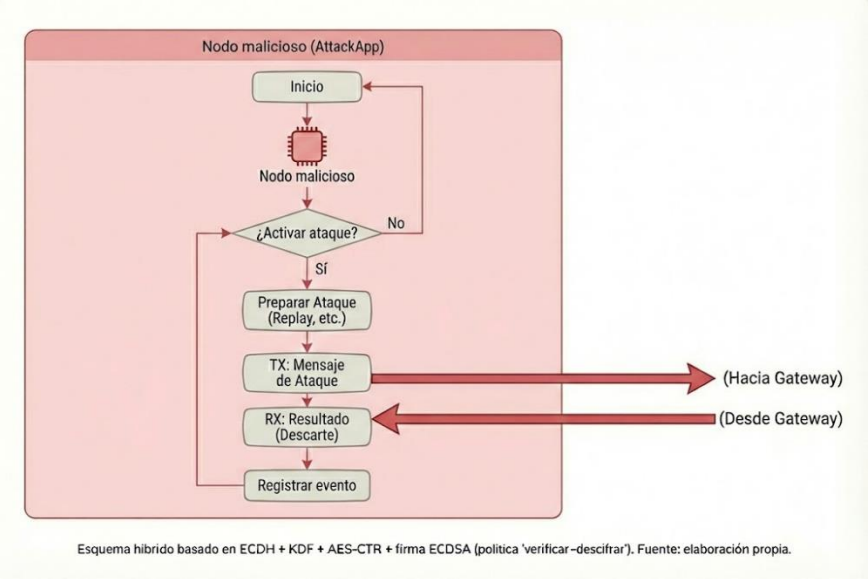


Figura 9

Diagrama de flujo del proceso criptográfico ECIES-based en OMNeT++: ECDH, derivación KDF, cifrado AES-CTR y verificación/firma ECDSA parte C, Atacante.



Esquema híbrido basado en ECDH + KDF + AES-CTR + firma ECDSA (política 'verificar–descifrar'). Fuente: elaboración propia.

3.3.1.1 Arquitectura general de la solución de seguridad

La solución de seguridad se integra como una transformación a nivel de aplicación sobre el payload, preservando invariante la pila base de comunicación y el canal acústico simulado. El diseño adoptado corresponde a un esquema híbrido basado en ECIES, donde el acuerdo de secreto mediante criptografía de curva elíptica permite derivar claves simétricas para cifrado eficiente, y la autenticidad e integridad se refuerzan mediante firma digital. En términos operativos, la arquitectura se compone de los siguientes elementos:

Bajo esta arquitectura, el flujo operativo general se resume de la siguiente forma:

(a) Identidad criptográfica por nodo (material de clave).

Cada entidad legítima (nodo sensor y gateway) dispone de un par de claves ECC utilizado para (i) establecer un secreto compartido y (ii) firmar/verificar mensajes. En el marco experimental, este material se inicializa de forma consistente por configuración, con el objetivo de mantener control del escenario y comparabilidad entre perfiles.

(b) Acuerdo de secreto y frescura por paquete.

El emisor y el receptor obtienen un secreto compartido mediante ECDH, del cual se deriva una clave simétrica para proteger el payload con AES en modo CTR. A diferencia de la formulación “ECIES formal” con par efímero por mensaje, la implementación experimental adopta pares de claves ECC preconfigurados/derivados de forma determinista para los nodos legítimos. La frescura por paquete se introduce mediante un IV aleatorio en AES-CTR, generado para cada mensaje, evitando reutilización de flujo y favoreciendo la observación del impacto de seguridad en tamaño y desempeño.

(c) Confidencialidad del payload (cifrado simétrico).

El contenido de aplicación se cifra con AES-CTR usando la clave derivada y el IV del mensaje, generando el ciphertext que se transporta en el campo de datos del paquete. El IV se adjunta como parámetro de descifrado, sin requerir modificaciones en el canal ni en la pila base.

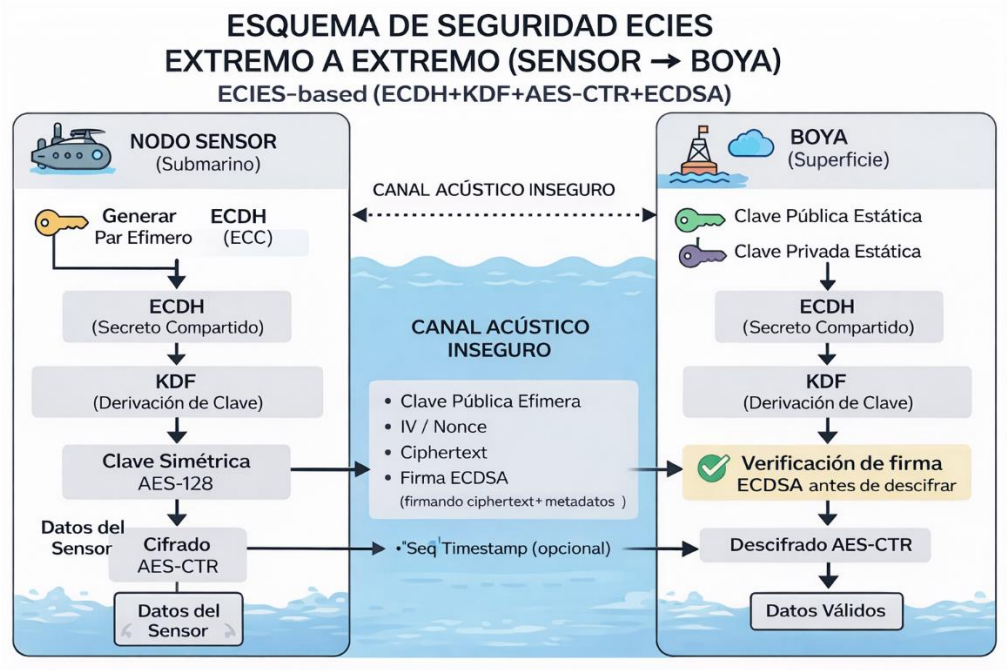
(d) Autenticidad e integridad (firma digital y política de aceptación).

El emisor genera una firma ECDSA asociada al contenido protegido, permitiendo al receptor validar integridad y autenticidad antes de procesar el mensaje. En la recepción se adopta una política operativa verify → decrypt, priorizando la verificación temprana (y el descarte en caso de fallo) para reducir procesamiento innecesario y mantener coherencia con el modelo de amenazas.

La **Figura 10** ilustra el esquema conceptual de seguridad ECIES extremo a extremo (Sensor → Boya) y la ubicación del bloque criptográfico dentro del flujo de comunicación.

Figura 10

Esquema extremo a extremo del envío seguro Sensor → Boya: ECDH + KDF → AES-CTR (IV por paquete) + firma ECDSA, con política verify→decrypt.



Este enfoque resulta adecuado para redes subacuáticas porque evita handshakes extensos en el medio acústico y concentra el costo principal en cómputo local (operaciones ECC + cifrado simétrico), lo cual puede ser más controlable que incrementar la señalización por el canal (Dini & Lo Duca, 2012; M. M. Islam et al., 2020). Adicionalmente, al tratarse de una solución modular, la arquitectura permite extender el diseño hacia topologías más complejas (por ejemplo, incorporación futura de nodos relay), manteniendo el mismo principio de seguridad end-to-end, en donde los nodos intermedios (si existieran) operarían únicamente como reenviadores sin acceso al contenido en claro.

3.3.2 Esquema criptográfico propuesto (ECIES / cifrado híbrido)

El módulo de seguridad se concibe como una unidad lógica con cuatro responsabilidades: (i) generación/gestión del material efímero (R), (ii) derivación de claves desde el secreto compartido mediante KDF, (iii) protección de confidencialidad mediante AES-CTR y (iv) autenticación e integridad mediante firma digital ECDSA (SHA-256)

calculada sobre el ciphertext y metadatos relevantes (Batool & Ahmad, 2021; Idris & Zukarnain, 2020; Securosys, s/f). El mensaje seguro resultante se encapsula junto con los metadatos mínimos requeridos (R e IV/nonce) para permitir al receptor verificar la firma, reconstruir las claves y recuperar el payload.

En la recepción, la verificación de la firma ECDSA constituye un paso previo obligatorio al descifrado, de modo que mensajes alterados o reinyectados sean descartados sin ser procesados por capas superiores.

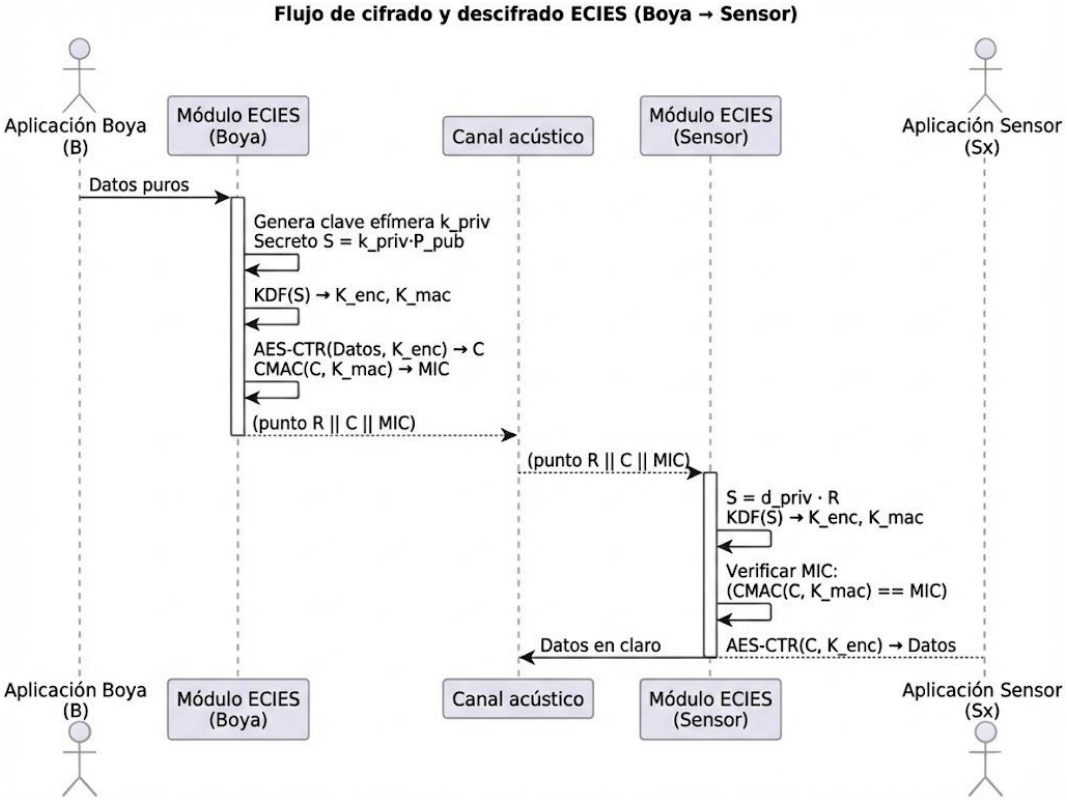
En este trabajo se adopta un esquema de cifrado híbrido basado en ECIES, donde el acuerdo de secreto (ECDH) y la derivación de claves (KDF) habilitan el cifrado simétrico eficiente (AES-CTR), y la autenticación/integridad se implementa mediante firma digital ECDSA bajo una política operativa verify → decrypt. Esta elección representa una adaptación orientada a UWSN, preservando el modelo de canal y facilitando la evaluación del overhead sin introducir handshakes adicionales.

Para la implementación se considera el uso de la curva secp256r1 (prime256v1), ampliamente utilizada en entornos prácticos y alineada a recomendaciones de estandarización para primitivas basadas en curvas elípticas (NIST, 2023a). La selección se justifica por el equilibrio entre nivel de seguridad, eficiencia computacional y tamaños de claves/puntos relativamente reducidos, lo cual es ventajoso en redes donde la transmisión de bytes adicionales implica costo significativo (Batool & Ahmad, 2021; Idris & Zukarnain, 2020).

La **Figura 11** presenta el flujo de cifrado y descifrado para el intercambio Boya → Sensor (análogo para Sensor → Boya). De forma resumida, el procedimiento operativo se describe como:

Figura 11

Flujo de ECIES (ECDH + KDF + AES-CTR) y autenticación mediante firma ECDSA entre Boya y Sensor



a) Fase de cifrado (emisor)

1. Generación de clave efímera: el emisor genera una clave privada efímera k_{priv}y su punto público asociado R = $k_{priv} \cdot G$.
2. Cálculo de secreto compartido (ECDH): se obtiene el secreto Smediante multiplicación escalar con la clave pública del receptor Q_R: S = $k_{priv} \cdot Q_R$.
3. Derivación de clave simétrica (KDF): a partir de S, una KDF (p. ej., basada en SHA-256) deriva la clave de cifrado K_{enc}(por ejemplo, 128 bits para AES-128). *(En este perfil, la integridad no se implementa con MAC, sino con firma digital; por tanto, no se requiere K_{aut} para autenticación).*
4. Cifrado de confidencialidad (AES-CTR): se genera un IV/nonce único por mensaje y se cifra el payload con AES-CTR usando K_{enc}, obteniendo el ciphertext C.

5. Autenticación e integridad (firma ECDSA): se calcula una firma ECDSA-SHA256 sobre Cy metadatos mínimos (p. ej., IV, identificador/contador anti-replay y/o tipo de mensaje).
6. Formato del paquete: el emisor transmite R || IV || C || Sig_{ECDSA} (y, si aplica, contador/ID de sesión) (y, si aplica, el contador/ID de sesión).

b) Fase de descifrado (receptor)

1. Reconstrucción del secreto: el receptor calcula S = $d_R \cdot R$, donde $d_R$es su clave privada y Rel punto efímero recibido.
2. Derivación de clave (KDF): aplica la misma KDF para obtener K_{enc}.
3. Verificación previa ("verify-then-decrypt"): verifica primero Sig_{ECDSA} sobre $(C, IV, metadatos)$. Si la firma es inválida, descarta el mensaje.
4. Descifrado (AES-CTR): si la firma es válida, descifra Ccon AES-CTR usando K_{enc}y el IV, y entrega el payload a la aplicación.

En el contexto UWSN, el valor agregado del enfoque híbrido es que la operación asimétrica (ECC) se utiliza principalmente para establecer el material de claves, mientras que el cifrado de datos se realiza de forma simétrica, reduciendo el costo computacional por paquete y permitiendo tasas de cifrado/descifrado compatibles con tráfico periódico de sensores (Batool & Ahmad, 2021; Goyal et al., 2022). Asimismo, al incorporar una firma (ECDSA-SHA256) la cual su valor generado con la clave privada del emisor sobre el ciphertext (y metadatos relevantes); permite verificar autenticidad e integridad en el receptor.

Se evita que la boya (o el sensor) procese como válidos paquetes alterados, lo que contribuye a mantener la integridad de la información incluso cuando el canal presenta pérdidas o colisiones (Dini & Lo Duca, 2012).

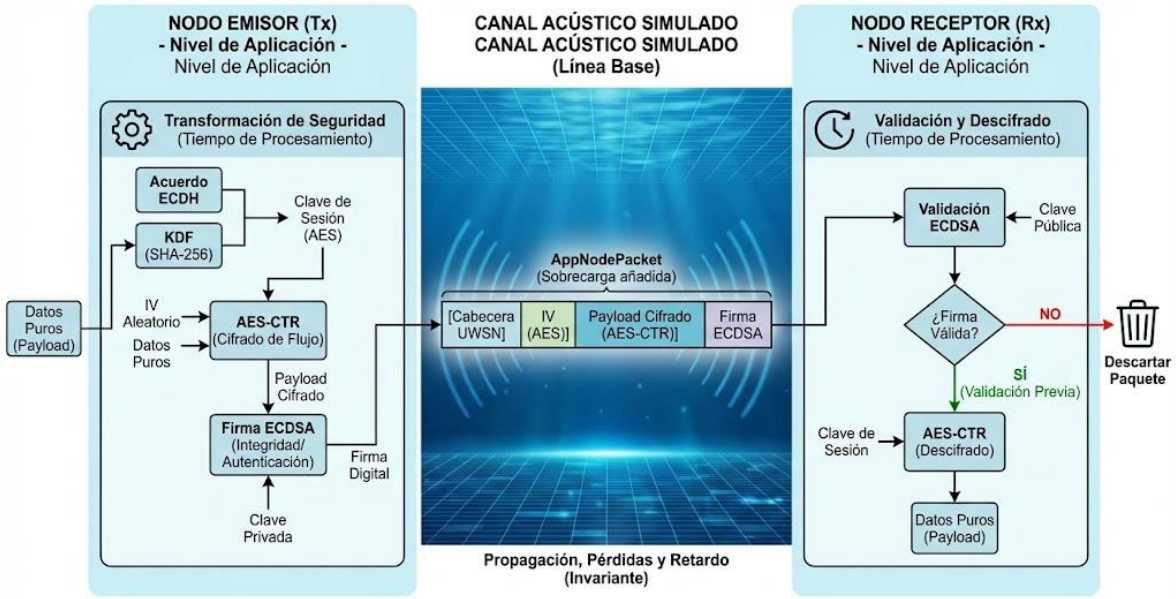
3.3.2.1 Adaptación del cifrado híbrido al entorno UWSN simulado

La adaptación del esquema híbrido tipo ECIES se fundamenta en mantener invariante el modelo base de comunicación acústica y concentrar el mecanismo de seguridad en el nivel de aplicación como una transformación del mensaje. Esta decisión permite que el canal, el retardo de propagación, las pérdidas y el acceso al medio permanezcan como línea base común, y que el impacto atribuible a seguridad se refleje únicamente en (i) la sobrecarga de campos añadidos al mensaje y (ii) el tiempo de procesamiento criptográfico en los nodos.

En la implementación del proyecto, el componente asimétrico se utiliza para derivar material de clave (acuerdo ECDH) y no para cifrar directamente el payload. Sobre el secreto compartido se aplica una derivación (KDF basada en hash) para obtener una clave simétrica de sesión, y el contenido se protege con un cifrado de flujo (AES en modo CTR) que introduce un IV aleatorio por mensaje. En paralelo, se incorpora autenticación e integridad mediante firma digital (ECDSA), permitiendo que el receptor ejecute validación previa antes de invertir cómputo en descifrado, en línea con el criterio de aceptación de la arquitectura. La **Figura 12** resume la estructura del encapsulado híbrido implementado (ECDH → KDF (SHA-256) → AES-CTR + firma ECDSA) y el empaquetado de campos dentro del mensaje AppNodePacket.

Figura 12

Estructura del encapsulado híbrido implementado (ECDH → KDF (SHA-256) → AES-CTR + firma ECDSA) y empaquetado de campos en AppNodePacket



3.3.2.2 Material criptográfico por nodo y representación en el mensaje

Para soportar identidad criptográfica en la simulación sin introducir infraestructura externa (p. ej., PKI completa), el proyecto deriva claves por nodo de forma determinística a partir de una frase configurada (parámetros gwpass y nodepass, con opción useIndexAsKey para diferenciar nodos). En particular, se obtiene una clave privada de 32 bytes y su clave pública asociada en formato comprimido (33 bytes) sobre la curva secp256r1, lo que reduce el tamaño de transmisión respecto a formatos no comprimidos y simplifica el intercambio de material público en mensajes.

En el formato de aplicación (AppNodePacket), los campos criptográficos se transportan como cadenas serializadas, priorizando compatibilidad de empaquetado en OMNeT++ y trazabilidad en la instrumentación. En modo seguro (S), el payload deja de circular en claro: el campo data representa el ciphertext, mientras que iv, publicKey y signature aportan los insumos necesarios para reconstrucción de clave de sesión, descifrado y validación de integridad/autenticidad. La **Tabla 23** resume los parámetros criptográficos implementados y su representación dentro del mensaje de aplicación.

Tabla 23

Parámetros criptográficos implementados y representación en el mensaje de aplicación.

Componente	Decisión implementada en el proyecto	Implicación experimental
Curva elíptica	secp256r1	Consistencia de tamaños y costo ECDH/ECDSA
Clave privada	32 bytes (derivada de passphrase)	Identidad por configuración; reproducible en simulación
Clave pública	33 bytes comprimida	Reduce overhead vs no comprimida
ECDH	Secreto compartido a partir de privada local + pública remota	Crea material de clave sin cifrado asimétrico directo
KDF	SHA-256 sobre secreto compartido; se toma material para AES	Coste fijo adicional; controlable
Cifrado simétrico	AES-CTR con IV aleatorio por mensaje	Unicidad por mensaje; costo por paquete
Firma	ECDSA+SHA-256 sobre el ciphertext (y campos asociados según flujo)	Habilita validación previa y contabilización de rechazos
Serialización	Campos en HEX dentro de strings (data/iv/publicKey/signature)	Aumenta tamaño del mensaje; hace visible overhead en medición

Nota: Descripción de la variante híbrida tipo ECIES implementada en el código mediante *ECDH para derivar clave + KDF + AES-CTR + ECDSA*.

3.3.3 Gestión de claves e identidad de nodos

La gestión de claves en la arquitectura propuesta se define bajo un modelo coherente con el alcance del trabajo: un entorno simulado donde se requiere asegurar comunicación de

datos sin desplegar una infraestructura completa de certificación. En este marco, se asume que:

a) Cada entidad (boya y sensor) posee un par de claves ECC de largo plazo (privada/pública) asociado a su identidad lógica.

b) La boya dispone de un repositorio local de claves públicas de los sensores autorizados (lista de confianza), y cada sensor dispone de la clave pública de la boya.

c) Las claves efímeras generadas por ECIES se utilizan por sesión/mensaje (según diseño), reduciendo el impacto de una eventual exposición posterior del material efímero, y aportando propiedades deseables como separación criptográfica entre transmisiones.

En términos de identificación, cada nodo sensor se representa mediante un identificador lógico (por ejemplo, NS^i), el cual permite seleccionar la clave pública correspondiente al momento de encapsular el mensaje. Esta relación identidad ↔ clave pública es crítica para evitar ataques de suplantación basados en sustitución de claves (key substitution). Por ello, en la arquitectura se considera que la distribución inicial de claves públicas se realiza por un canal confiable fuera de banda (por ejemplo, precarga de parámetros del escenario de simulación), lo que es consistente con modelos propuestos en seguridad para UWSN en los que la autenticación basada en ECC se apoya en material criptográfico preestablecido para reducir señalización en el canal acústico (M. M. Islam et al., 2020; Wahid et al., 2021).

Como aspecto de diseño orientado a robustez, y alineado a recomendaciones comunes para evitar reutilización de mensajes, se contempla el uso de mecanismos de frescura (por ejemplo, contador/nonce), los cuales pueden incorporarse como parte de los metadatos autenticados del mensaje. Esta medida es relevante para reducir el riesgo de ataques por

repetición (replay) en enlaces donde el atacante pueda capturar y retransmitir paquetes previamente observados (Dini & Lo Duca, 2012).

3.3.3.1 Identidad criptográfica y alcance dentro del entorno simulado

En el sistema propuesto, la identidad de cada entidad participante (nodos sensores y gateway) se representa mediante un par de claves elípticas asociado al rol del módulo de aplicación. Esta identidad permite dos funciones fundamentales dentro del flujo seguro: (i) habilitar la derivación de un secreto compartido para protección de confidencialidad mediante un esquema híbrido, y (ii) permitir autenticación e integridad mediante firma digital. En términos de arquitectura, la identidad criptográfica se vincula al nodo como un atributo lógico de la aplicación, preservando el canal acústico y la pila base de comunicación como componentes invariantes del escenario experimental.

Bajo este enfoque, el gateway mantiene un rol de referencia estable: su identidad criptográfica representa el punto de convergencia hacia el cual se dirigen los sensores. De manera complementaria, cada sensor conserva una identidad propia que le permite generar mensajes firmados y recuperar claves de sesión derivadas, de forma consistente con el modelo de amenaza y con la política de aceptación basada en verificación previa.

3.3.3.2 Inicialización determinística de claves para control experimental

Con el propósito de garantizar repetibilidad y control experimental, el material criptográfico se inicializa de forma determinística a partir de parámetros de configuración del escenario. En la implementación, tanto el gateway como los nodos derivan su clave privada como función de una frase configurada (gwpass para el gateway y nodepass/identificador para el nodo). Esta estrategia evita dependencias externas de provisión de llaves y reduce variabilidad no controlada entre corridas, permitiendo que los cambios observados se atribuyan al perfil de seguridad y a la carga/escala del escenario.

Para diferenciar identidades entre nodos, el proyecto incorpora una política opcional en la que el índice del nodo se incorpora como semilla de derivación (useIndexAsKey), evitando que múltiples sensores compartan la misma identidad criptográfica cuando se requiere escalamiento. Adicionalmente, la clave pública se representa en formato comprimido, reduciendo el tamaño de transmisión y haciendo explícito el overhead cuando se incorpora en el mensaje seguro. La **Tabla 24** resume los parámetros de configuración asociados a identidad y gestión de claves utilizados en la simulación.

Tabla 24

Parámetros de configuración asociados a identidad y gestión de claves en la simulación.

Parámetro	**Ubicación (modelo/configuración)**	**Propósito**	**Impacto en la evaluación**
gwpass	Parámetro del módulo de aplicación (Gateway)	Semilla para derivar identidad criptográfica del gateway	Repetibilidad; referencia estable para encapsulado/validación
nodepass	Parámetro del módulo de aplicación (Sensor)	Semilla para derivar identidad criptográfica de nodos cuando no se usa índice	Controlado; permite escenarios con identidad común (si se desea)
useIndexAsKey	Parámetro del módulo (Sensor)	Diferencia identidades por índice del nodo	Evita colisiones de identidad al escalar densidad

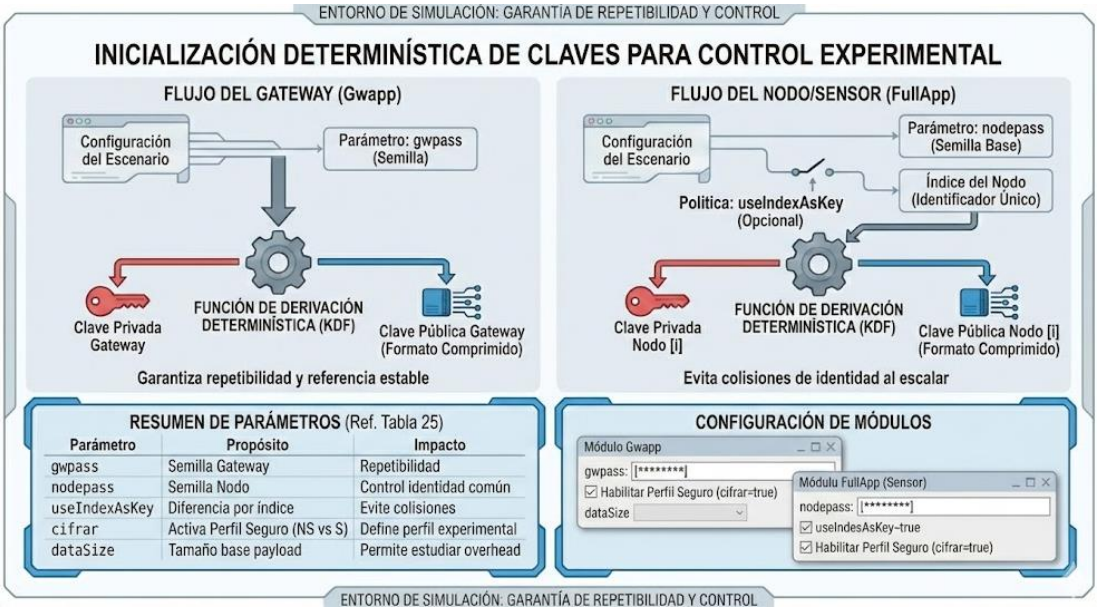
cifrar	Parámetro del módulo (Gateway y Sensores)	Activa/desactiva el perfil seguro (NS vs S)	Define el perfil experimental sin alterar el canal
dataSize	Parámetro del módulo (Gateway y Sensores)	Tamaño base del payload lógico	Permite estudiar overhead relativo por campos criptográficos

La habilitación del perfil seguro y la política de identidad se controlan mediante parámetros de configuración a nivel de aplicación, permitiendo activar o desactivar el cifrado sin modificar el modelo base del canal ni la lógica de tráfico. La

Figura 13 muestra la selección de parámetros utilizada para habilitar cifrado y gestión de claves en los módulos FullApp y Gwapp.

Figura 13

Selección de parámetros de configuración para habilitar cifrado y gestión de claves en FullApp y Gwapp.



3.3.3.3 Disponibilidad de claves públicas durante la ejecución (sin infraestructura externa)

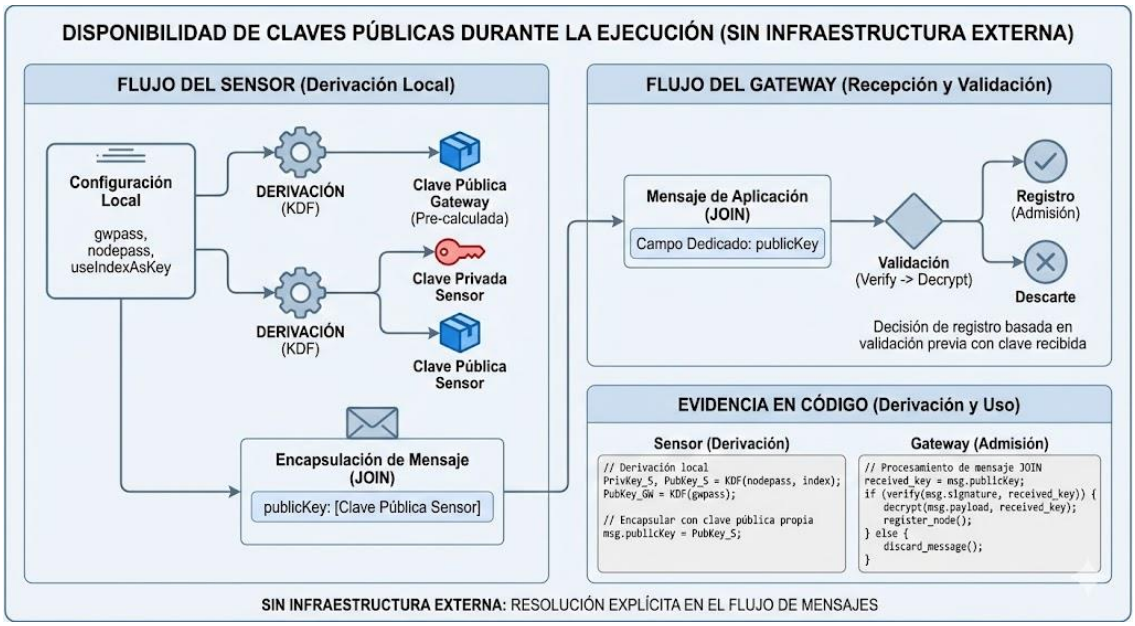
En la ejecución del perfil seguro, el intercambio de información pública se resuelve de forma explícita dentro del propio flujo de mensajes de aplicación. En el caso de los sensores, el nodo deriva localmente (a partir de la configuración) tanto su clave privada como su clave pública y, además, obtiene la clave pública del gateway mediante la misma política de derivación asociada a gwpass. Con ello, el sensor cuenta con el insumo necesario para encapsular mensajes dirigidos al gateway sin requerir un proceso adicional de consulta o distribución dinámica.

Desde la perspectiva del gateway, la clave pública del sensor se incorpora en el mensaje de aplicación como campo dedicado (publicKey). Este diseño permite que el gateway verifique firmas y derive secretos compartidos bajo el supuesto de que la clave pública recibida pertenece al emisor legítimo, hipótesis coherente con el marco experimental y con el modelo de amenazas utilizado para evaluar manipulación del tráfico. En el flujo de admisión (JOIN), esta disponibilidad de la clave pública es crítica para aplicar la validación

previa y decidir registro o descarte. La **Figura 14** presenta evidencia directa en código de (i) la derivación determinística de claves y (ii) el uso del campo *publicKey* para habilitar verificación y descifrado condicionado (verify → decrypt) durante la admisión y entrega.

Figura 14

Esquema de gestión de claves en tiempo de ejecución de la derivación determinística y disponibilidad de clave pública mediante el campo publicKey (Clave Pública).



Nota. El esquema detalla el ciclo de seguridad implementado en los módulos *FullApp* y *Gwapp*, iniciando con la derivación local de credenciales a partir de parámetros de configuración para garantizar repetibilidad. Durante la comunicación, el sensor integra su clave pública en el mensaje cifrado con AES-CTR y firmado mediante ECDSA, permitiendo que el gateway realice la extracción de la identidad, la validación de la firma y el descifrado del *payload* en un único flujo de procesamiento autónomo, sin requerir infraestructura de gestión externa.

Para evitar sustitución de claves (key substitution) en el entorno simulado, se asume que el gateway dispone de una lista de claves públicas autorizadas (whitelist) precargada por configuración. En consecuencia, cuando un mensaje incluye el campo publicKey, el gateway

no confía ciegamente en dicho valor: lo compara contra la clave pública registrada para la identidad lógica del nodo (por ejemplo, su índice o identificador). Solo si existe correspondencia, se procede con la verificación de firma y, posteriormente, con la derivación/descifrado.

3.3.3.4 Uso operativo de la identidad en el flujo seguro

Una vez inicializada la identidad criptográfica, el uso operativo sigue un patrón consistente: el emisor encapsula el payload derivando una clave de sesión a partir de un secreto compartido (acuerdo ECDH) y cifra el contenido con AES-CTR empleando un IV por mensaje. En paralelo, genera una firma digital que permite autenticidad e integridad del contenido protegido. En la recepción, el nodo recupera la clave pública del emisor (desde el mensaje o desde el rol configurado) para verificar la firma y, cuando corresponde, deriva el secreto compartido con su clave privada local para descifrar.

En el proceso de admisión (JOIN), la identidad criptográfica se integra directamente a la política de aceptación: el receptor aplica validación previa y únicamente registra/acepta al nodo si el contenido recuperado corresponde a un mensaje válido de incorporación. Este comportamiento refuerza el control de acceso bajo tráfico adversarial, y además ofrece trazabilidad experimental mediante eventos de aceptación, rechazo y listas de bloqueo.

3.3.3.5 Registro, descarte y listas de control como parte de la gestión de identidad

Como parte de la operación segura, el sistema mantiene estructuras lógicas asociadas a identidad: conjunto/lista de nodos admitidos y una lista de bloqueo (blacklist) para emisores que presentan comportamiento inválido o persistente durante la fase de incorporación. Bajo el perfil seguro, la blacklist actúa como mecanismo defensivo ante intentos repetidos de admisión o mensajes con validación fallida, reduciendo el costo de procesamiento posterior y contribuyendo a estabilizar el comportamiento en presencia de atacantes activos.

La detección operativa de emisores no admitidos se apoya en el control de admisión criptográfica. En particular, si una solicitud de JOIN no supera el criterio esperado de validación y recuperación del contenido (por ejemplo, verificación satisfactoria y/o descifrado coherente con el mensaje de admisión), el gateway registra al emisor en una lista de bloqueo (blacklist) y descarta su tráfico posterior. Este mecanismo no pretende modelar un sistema de detección de intrusos (IDS) basado en tasa, ráfagas o análisis estadístico del tráfico, sino un control mínimo coherente con el alcance experimental: separar nodos legítimos de emisores no autorizados bajo el esquema criptográfico integrado y permitir que los efectos de saturación/DoS se reflejen como degradación del desempeño de comunicación en condiciones controladas.

3.3.4 Modelo de amenazas

El modelo de amenazas establece las capacidades del atacante y delimita los mecanismos de protección esperados. Para la arquitectura propuesta se asume un atacante con acceso al medio acústico (atacante de canal), capaz de:

a) Interceptar mensajes transmitidos (escucha pasiva/eavesdropping).

b) Reproducir mensajes capturados (replay).

c) Inyectar paquetes falsos y/o modificar paquetes en tránsito (manipulación activa).

d) Intentar suplantación de identidad, presentándose como un nodo legítimo ante la boya o ante un sensor.

Bajo este modelo, la arquitectura busca mitigar principalmente amenazas de confidencialidad e integridad. En primer lugar, el uso de ECIES con confidencialidad con AES-CTR + integridad/autenticación con firma ECDSA impide que un atacante pasivo obtenga el contenido del mensaje sin disponer de la clave privada legítima del receptor. En segundo lugar, la verificación de la firma ECDSA permite detectar modificaciones o

falsificaciones de paquetes, lo que reduce la probabilidad de aceptación de información alterada, coherente con suites de comunicación segura propuestas para redes acústicas subacuáticas.

Adicionalmente, la prevención robusta de ataques tipo man-in-the-middle (MITM) depende de la confiabilidad del mecanismo de distribución/validación de claves públicas. En el alcance planteado, al asumir precarga confiable del material de claves públicas, se reduce de forma práctica el espacio de ataque para MITM basado en sustitución de claves. No obstante, se reconoce que en despliegues reales se requeriría un mecanismo formal de gestión de confianza (por ejemplo, certificación, identidad criptográfica o protocolos de autenticación avanzados) para fortalecer esta propiedad, tal como se discute en investigaciones recientes sobre autenticación en UWSN (M. M. Islam et al., 2020; Luo et al., 2023b).

Finalmente, amenazas como denegación de servicio (DoS) por interferencia deliberada (jamming), incremento artificial de colisiones, o captura física de nodos no se resuelven únicamente con criptografía y, en un entorno simulado, se consideran fuera del alcance principal de la propuesta. Sin embargo, la arquitectura permite futuras extensiones hacia esquemas complementarios (p. ej., detección de intrusos o autenticación reforzada), en concordancia con tendencias de seguridad e IDS aplicadas a entornos IoT y redes especializadas (M. Khan & Kim, 2022; Rahman & Kim, 2022).

3.3.4.1 Supuestos del atacante y capacidades consideradas

El modelo de amenaza adoptado representa un atacante con acceso al medio acústico, capaz de observar transmisiones y, dependiendo del perfil experimental, interferir activamente en el tráfico. En coherencia con un enfoque de evaluación de seguridad a nivel de protocolo y operación del sistema, se asume que el atacante puede capturar paquetes completos en tránsito (incluyendo campos de control y campos criptográficos) y, cuando se

habilita el perfil de ataque activo, puede también inyectar o alterar mensajes para perturbar la comunicación.

Sin embargo, el modelo no asume capacidades de ruptura criptográfica. En particular, no se considera que el atacante tenga acceso a material secreto de configuración interna (por ejemplo, semillas de derivación utilizadas para construir claves privadas en la simulación), ni que disponga de recursos o avances que le permitan vulnerar primitivas estándar (ECDH/ECDSA/AES). Bajo estas condiciones, el sistema se evalúa frente a amenazas realistas de canal (escucha, inyección y manipulación), y se mide el impacto del mecanismo de seguridad sobre aceptación/descartes y desempeño.

3.3.4.2 Atacante pasivo (Listener): qué observa y por qué no puede derivar claves

El atacante pasivo se modela como un observador del canal que puede capturar el contenido transmitido, incluyendo los campos del mensaje seguro. En este escenario, el atacante puede "ver" (i) el ciphertext transportado en el payload protegido, (ii) el IV/nonce usado por AES-CTR, (iii) la firma ECDSA, y (iv) la clave pública enviada en el propio mensaje (por ejemplo, el campo publicKey del emisor) o cualquier otro dato público de identificación.

Que el atacante pueda observar claves públicas no implica que pueda derivar la clave de sesión. La razón es estructural: el material que permite cifrar/descifrar no se obtiene de información pública, sino del secreto compartido derivado mediante ECDH, el cual requiere al menos una clave privada legítima para ser calculado. Aunque el atacante capture ambas claves públicas (emisor y receptor), la reconstrucción del secreto compartido sin la clave privada equivale a resolver un problema criptográfico considerado intratable en el modelo de seguridad de curvas elípticas (supuesto estándar de dureza). Por ello, el atacante puede observar el intercambio, pero no puede reconstruir la clave simétrica derivada por la KDF ni descifrar el ciphertext.

De forma análoga, el atacante puede verificar firmas (la verificación usa claves públicas), pero no puede fabricar una firma válida para suplantar a un emisor legítimo sin poseer la clave privada correspondiente. Esto refuerza el objetivo del perfil seguro: incluso bajo escucha total del canal, la confidencialidad del payload se mantiene, y la autenticidad/integridad impide que el atacante pasivo "convierta" su observación en capacidad de falsificación.

3.3.4.3 Atacante activo (Atackapp): acciones modeladas y límites asumidos

El atacante activo se modela con capacidades de canal orientadas a degradar el servicio y provocar aceptación de tráfico inválido: (i) inyección de mensajes con campos arbitrarios, (ii) alteración de paquetes capturados (modificación de payload o metadatos), y (iii) perturbación del patrón de envío para incrementar colisiones o saturación. Bajo este modelo, el atacante intenta que el receptor procese mensajes no legítimos o que consuma recursos en tráfico inválido.

No obstante, el atacante activo no se modela con capacidad de generar firmas válidas sin claves privadas legítimas ni de derivar claves de sesión por observación. En consecuencia, sus intentos de suplantación o manipulación quedan acotados por el mecanismo de validación: cualquier modificación del ciphertext o de campos firmados conduce, con alta probabilidad, a fallos de verificación y descarte. Este comportamiento es precisamente el que se busca someter a evaluación: la efectividad del esquema para transformar intentos de manipulación en eventos de rechazo observables.

3.3.4.4 Relación del modelo de amenaza con la política verify → decrypt y la instrumentación

La política verify → decrypt se adopta para proteger recursos del nodo ante tráfico inválido: el receptor verifica autenticidad e integridad mediante ECDSA antes de descifrar, descartando tempranamente mensajes alterados o reinyectados. Esta secuencia reduce

procesamiento innecesario y permite instrumentar eventos de rechazo como variable observable de seguridad en simulación.

Esto habilita mediciones específicas en el entorno simulado: (i) tasa de verificación fallida, (ii) descartes atribuibles a seguridad, (iii) impacto del atacante sobre entrega y latencia, y (iv) diferencias entre perfiles NS y S bajo escucha o intervención activa. Estas salidas estructuran el puente hacia el Capítulo 4, donde se cuantifica el efecto del atacante bajo variaciones controladas de densidad y carga.

3.4 Diseño del entorno de simulación en OMNeT++

Esta sección describe el entorno de simulación utilizado para modelar una red de sensores submarinos (UWSN) y evaluar el impacto de integrar un esquema criptográfico basado en ECIES dentro del flujo de comunicación. El propósito del capítulo no es documentar el código, sino definir el marco experimental: plataforma, modelos base utilizados, organización de escenarios, supuestos del medio acústico y mecanismos de instrumentación para recolectar métricas bajo configuraciones reproducibles. La **Figura 15** sintetiza el modelo de referencia de comunicación subacuática empleado como línea base, la capa de evaluación que parametriza los escenarios, y el flujo de extracción de métricas utilizado para el análisis comparativo.

Figura 15

Modelo de referencia de comunicación subacuática y capa de evaluación experimental para construcción de escenarios y extracción de métricas.

3.4.1 Plataforma de simulación y componentes del entorno

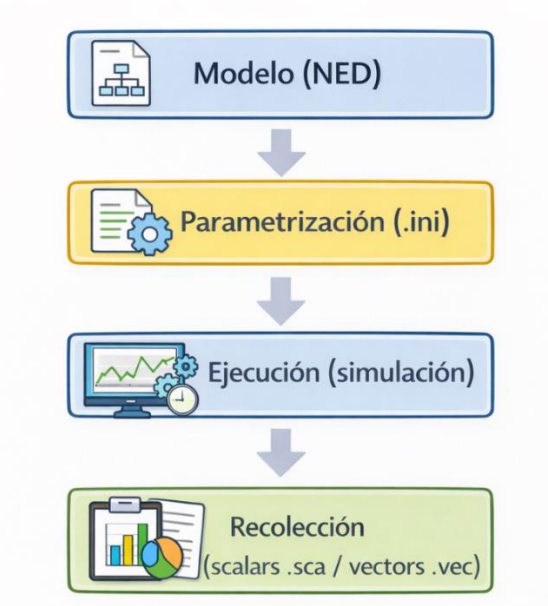
La evaluación experimental se ejecutó sobre OMNeT++ 6.1, seleccionado por su capacidad de simulación de eventos discretos y por permitir instrumentación reproducible mediante control de parámetros y semillas. El entorno se organizó en dos capas: (i) un modelo base, responsable del comportamiento del canal acústico y la conectividad; y (ii) una capa de evaluación, encargada de definir escenarios (6 condiciones: NS/S × Normal/Atacante/Listener) e instrumentación de métricas. Esta separación reduce variabilidad no controlada en las comparaciones, ya que el canal y la pila inferior permanecen invariantes mientras el factor experimental se limita a la incorporación de seguridad y/o atacante. La

Figura 16 resume esta organización conceptual y la relación entre modelado, parametrización experimental y recolección de resultados.

En términos prácticos, estos componentes se materializan como dos proyectos separados dentro del workspace (denominados *UanModel-master* para el modelo base y *Prueba* para la capa experimental). Sin embargo, a lo largo del documento se emplea la nomenclatura modelo base y capa de evaluación, por corresponder al rol metodológico que cumplen en el diseño experimental.

Figura 16

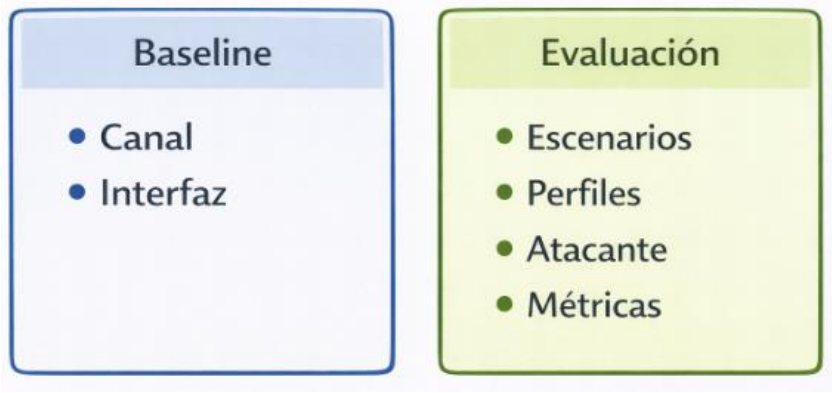
Organización conceptual del entorno, separando modelado, parametrización y recolección de resultados.



La **Figura 17** presenta la organización conceptual del entorno, diferenciando el modelo base de la capa de evaluación, e incorporando las etapas de parametrización experimental y recolección de resultados.

Figura 17

Organización conceptual del entorno, separando el modelo base de la capa de evaluación, así como las etapas de parametrización



3.4.2 Modelos reutilizados y componentes activos del escenario Dr. Alfonso Ariza

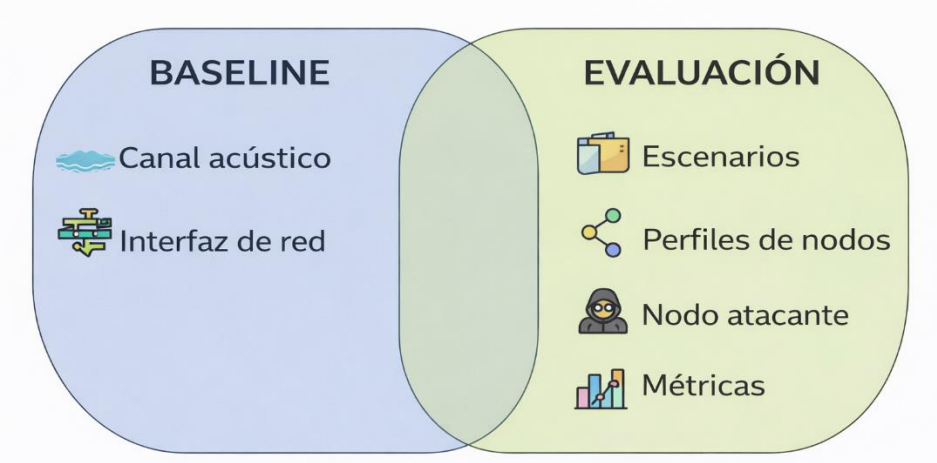
Se nos ha proporcionado una herramienta base fundamental heredada por el Dr. Alfonso Ariza el cual a través de su investigación (Ariza, 2026) se obtiene el modelo base de "Uanmodel-Master" cual tomamos como plantilla principal y extracción de los módulos necesarios para el funcionamiento del entorno de simulación. El modelo base aporta los elementos funcionales necesarios para representar la comunicación acústica submarina, incluyendo el medio de propagación y la interfaz empleada por los nodos para acceder al canal. Esta base permite disponer de un comportamiento consistente del canal y de la pila mínima de comunicación, evitando introducir variaciones no controladas al momento de comparar escenarios.

Sobre este baseline, la capa de evaluación define las redes experimentales y agrega componentes orientados a la medición, tales como: perfiles de ejecución, mecanismos de activación de seguridad, y la inclusión de un nodo atacante en aquellos escenarios donde corresponde. Esta estrategia favorece la reutilización y asegura que la comparación entre ejecuciones se realice bajo el mismo modelo subyacente de canal, variando únicamente los factores experimentales definidos (densidad, seguridad y/o atacante). La

Figura 18 resume, a nivel conceptual, qué elementos pertenecen al modelo base y cuáles son introducidos por la capa de evaluación para construir escenarios comparables.

Figura 18

Diagrama simple de "baseline" vs "evaluación" con bullets (canal/interfaz vs escenarios/perfiles/atacante/métricas).



3.4.2.1 UanSoundMedium como núcleo del canal acústico simulado

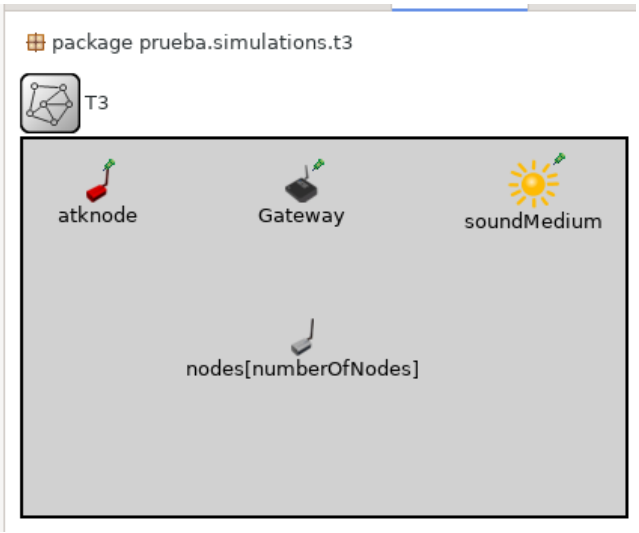
El escenario experimental se construye sobre un modelo de canal explícito representado por el submódulo soundMedium: UanSoundMedium, es un trabajo hererado por (Ariza, 2026) el cual se ha trabajado sobre el, instanciado a nivel de red (p. ej., en los escenarios T2/T3/T4). Este componente constituye el medio compartido de la simulación acústica: centraliza las condiciones de propagación y actúa como punto de referencia para que las interfaces submarinas (NIC UAN) evalúen si una transmisión es recibida, con qué degradación y bajo qué condiciones de interferencia. En términos prácticos, el canal no se "distribuye" dentro de cada nodo; se declara como un medio común del escenario, lo que favorece comparabilidad y control experimental.

En el modelo de nodo, la interfaz UanCsmaCaInterface se conecta con la aplicación mediante sockets (upperLayerIn/upperLayerOut), preservando una pila consistente de extremo a extremo: la aplicación genera el mensaje (en perfil NS o S), la NIC resuelve el acceso al medio y el comportamiento de enlace, y UanSoundMedium gobierna la entrega física sobre el canal acústico. Esta separación es metodológicamente útil porque permite aislar el impacto de seguridad como transformación del payload a nivel de aplicación, sin

alterar el comportamiento del medio ni el mecanismo de acceso al canal. La **Figura 19** ilustra la topología del escenario y la declaración del medio acústico compartido.

Figura 19

Topología del escenario y declaración del medio acústico UanSoundMedium.



Nota: Se observa el submódulo soundMedium como entidad compartida del canal y los nodos conectados mediante su interfaz UAN.

3.4.2.2 Componentes activos del stack y su relación con INET

El escenario reutiliza componentes de infraestructura provistos por INET (por ejemplo, movilidad estacionaria y tablas de interfaz) para mantener una base estable de simulación y aprovechar mecanismos ya validados para inicialización, direccionamiento y conexión entre módulos. Sobre esa base, la comunicación subacuática se concreta mediante el stack UAN (interfaz UanCsmaCaInterface y el medio UanSoundMedium), que introduce el comportamiento propio de un canal acústico compartido sin requerir que el proyecto redefina desde cero la física del enlace.

En este diseño, INET aporta elementos transversales (como InterfaceTable y StationaryMobility), mientras que el paquete UAN aporta los elementos específicos del

dominio submarino (canal acústico y NIC asociada). Esta combinación reduce variación no controlada y permite centrar el análisis del Capítulo 4 en el efecto de seguridad y de escala (densidad/carga), en lugar de atribuir diferencias a cambios en modelos base de red.

3.4.2.3 Importancia de FLoRA en el proyecto (dependencia y trazabilidad del entorno)

Aunque el canal acústico y la NIC del experimento se modelan con componentes UAN, el workspace del proyecto incorpora FLoRA como dependencia del entorno. En la estructura actual, dicha dependencia aparece a nivel de definiciones NED (por ejemplo, importación de flora.LoRaPhy.LoRaMedium en algunos escenarios), lo cual implica que FLoRA forma parte del conjunto de proyectos necesarios para compilar y resolver referencias NED dentro del workspace, aun cuando el medio LoRa no sea instanciado como canal activo en las ejecuciones UWSN.

Desde la perspectiva metodológica, esta distinción es relevante: FLoRA no reemplaza el medio acústico ni participa como canal de propagación en los perfiles UWSN; su contribución se asocia a la reutilización del entorno de simulación y a la consistencia del workspace (resolución de paquetes/imports y soporte de módulos heredados o prototipos). En consecuencia, el proyecto distingue entre (i) dependencias del workspace requeridas para construcción/compilación y (ii) componentes efectivamente activos en la simulación del canal acústico. **Tabla 25** resume las dependencias del entorno (workspace) y su participación real dentro de la simulación UWSN.

Tabla 25

Dependencias del entorno (workspace) y participación en la simulación UWSN.

Componente	**Rol en el proyecto**	**¿Activo en runtime UWSN (T2/T3)?**	**Justificación**
INET	Infraestructura base (movilidad, tablas, utilidades)	Sí	Base estable y ampliamente usada para composición de nodos
UAN (UanSoundMedium, UanCsmaCaInterface)	Canal acústico + NIC submarina	Sí	Medio y acceso al canal del experimento UWSN
FLoRA	Dependencia del workspace / referencias NED / reutilización	No (en canal acústico)	Requerido para resolver imports NED y mantener compatibilidad del proyecto

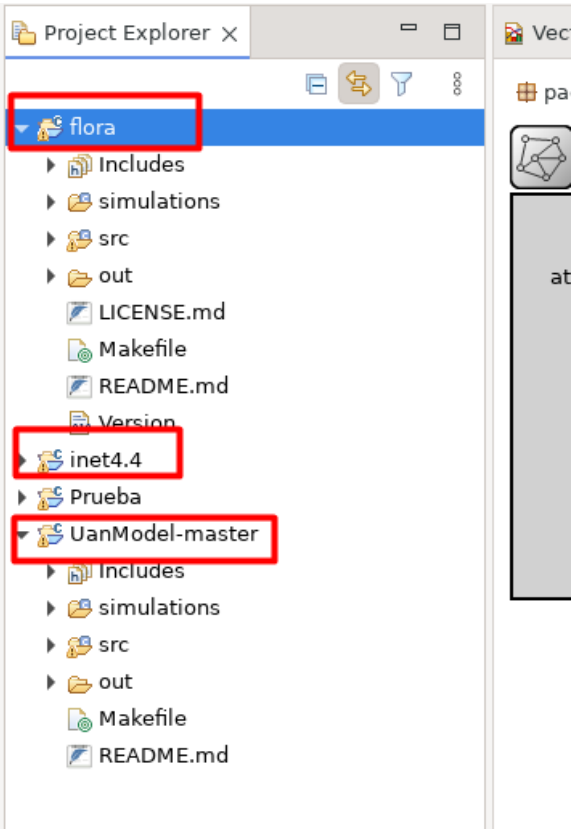
Como se resume en la **Tabla 25**, el entorno de simulación se apoya en INET y UAN como componentes activos del modelo UWSN, mientras que FLoRA se mantiene como una dependencia del workspace para resolver referencias e imports NED durante la compilación, sin actuar como medio de propagación en los escenarios acústicos.

En este sentido, la

Figura 20 muestra en el Project Explorer los proyectos cargados (FLoRA, INET y UanModel-master), evidenciando la trazabilidad de dependencias requerida por el entorno de trabajo.

Figura 20

Proyectos cargados en el workspace de OMNeT++ (FLoRA, INET y UanModel-master) para resolución de dependencias NED.



3.4.2.4 Disciplina temporal de transmisión: control MAC CSMA/CA y flujo de peticiones de envío

La comunicación acústica simulada se rige por un mecanismo de acceso al medio basado en contención (CSMA/CA), el cual limita la simultaneidad de transmisión y determina el instante efectivo en el que un paquete puede ocupar el canal compartido. En este

enfoque, la aplicación no transmite de forma inmediata: genera el mensaje y solicita su envío, pero la transmisión queda sujeta a la decisión del enlace, que administra colas y temporizadores antes de habilitar el acceso al canal.

Desde la perspectiva funcional, el flujo se organiza como clasificación y encolamiento (classifier/queue), seguido del control de acceso en el módulo MAC, y finalmente la entrega hacia la radio para la emisión. El MAC aplica una política de contención típica de CSMA/CA, que incluye verificación de disponibilidad del canal (CCA) y tiempos de espera/backoff, utilizando una granularidad temporal (*slotTime*) para sus temporizadores internos. En consecuencia, aunque la aplicación solicite el envío en un instante dado, el paquete puede permanecer en cola hasta que el MAC determine que el medio está disponible y se cumplan las condiciones de acceso.

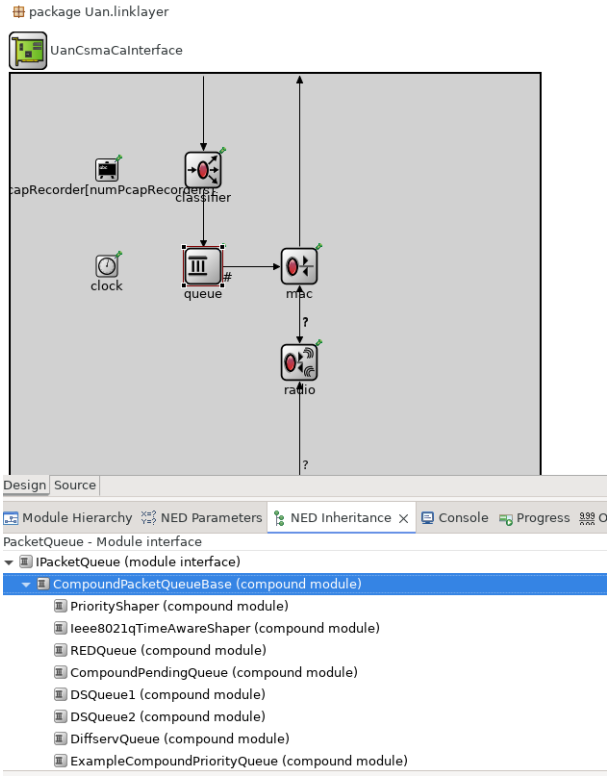
Esta característica es relevante para la interpretación del desempeño, ya que el retardo extremo a extremo y la tasa efectiva dependen no solo del tamaño del mensaje (y del overhead criptográfico), sino también del tiempo de espera inducido por contención y encolamiento. En esta tesis, el mecanismo de acceso al medio se mantiene constante entre perfiles comparables (NS y S), de modo que las diferencias medidas se atribuyan principalmente al encapsulado de seguridad y a su costo de procesamiento, sin introducir cambios en el comportamiento del enlace.

Como se ilustra en la

Figura 21, el flujo de transmisión se encola y queda sujeto a la política temporal del enlace: el módulo MAC (CSMA/CA) gobierna el momento efectivo de transmisión mediante temporización y control de contención (CCA/backoff), mientras que la aplicación únicamente solicita el envío.

Figura 21

Estructura interna de la interfaz UanCsmaCaInterface en el stack UAN.



Nota: el tráfico de aplicación ingresa a una cola (queue) y su transmisión es autorizada por el módulo MAC (CSMA/CA) en coordinación con el radio y la temporización del enlace (clock).

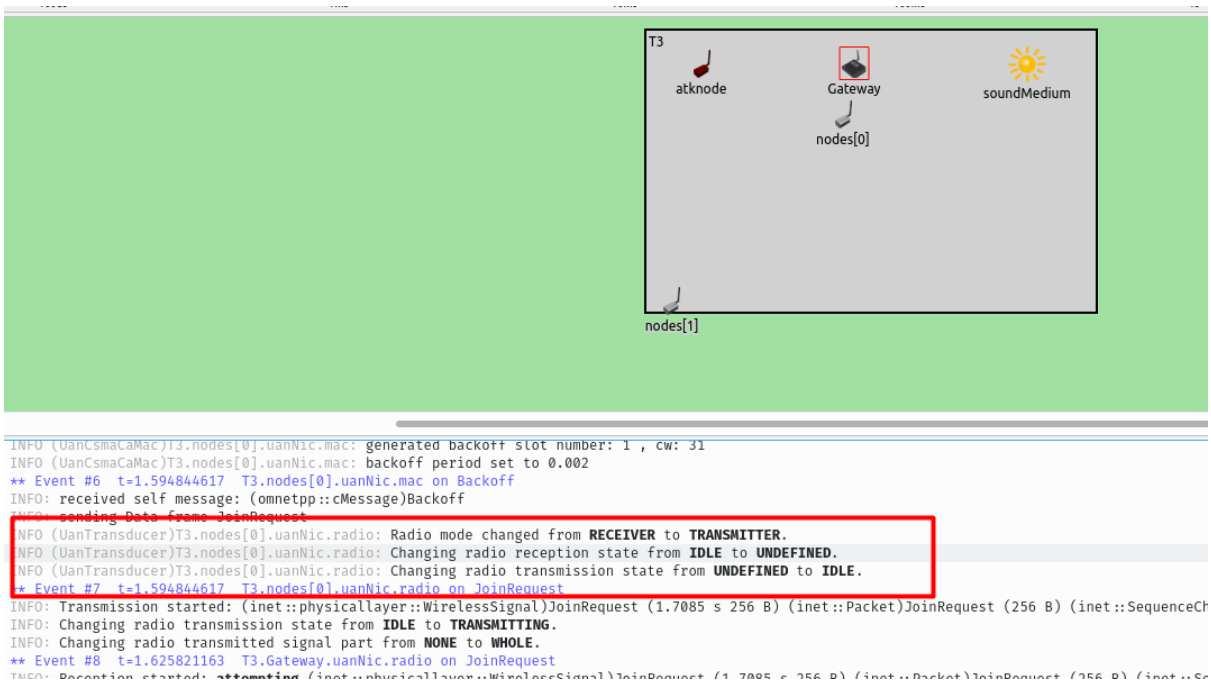
3.4.2.5 Operación half-duplex y su impacto en la comunicación

El modelo de enlace asume operación half-duplex, es decir, un nodo no transmite y recibe de manera simultánea en el mismo intervalo operativo. Esta restricción es coherente con la naturaleza de sistemas acústicos y con la disciplina por timeslots: durante una transmisión, el nodo suspende la recepción, y viceversa. Como consecuencia, intercambios que requieren respuesta (por ejemplo, JOIN seguido de JOIN_REPLY) incorporan tiempos de espera naturales para conmutar entre estados y acceder nuevamente al medio, lo cual incrementa el retardo efectivo y hace que la contención bajo densidad sea un factor crítico.

En el marco experimental, esta propiedad se mantiene constante entre perfiles NS y S; por tanto, cualquier variación adicional observada en resultados se interpreta como efecto del encapsulado seguro (sobrecarga de campos y costo de procesamiento) y no como un cambio en la capacidad dúplex del enlace, Esto se puede evidenciar en la **Figura 22** la cual muestra el canal acústico y su operación en half-duplex.

Figura 22

Evidencia de canal acústico común y operación half-duplex en el escenario T3 (UanSoundMedium y cambios de modo del radio).



3.4.3 Modelo de nodo y punto de inserción del mecanismo criptográfico

El mecanismo de seguridad se incorporó a nivel de aplicación, de manera que el payload se protege antes de ingresar al canal acústico y se recupera únicamente en el destino legítimo. Operativamente, esto permite que la red se mantenga comparable entre perfiles, ya que los modelos de propagación y acceso al medio no se alteran; las diferencias observadas en métricas se atribuyen principalmente a la sobrecarga (tamaño del mensaje) y al costo de procesamiento criptográfico en los nodos.

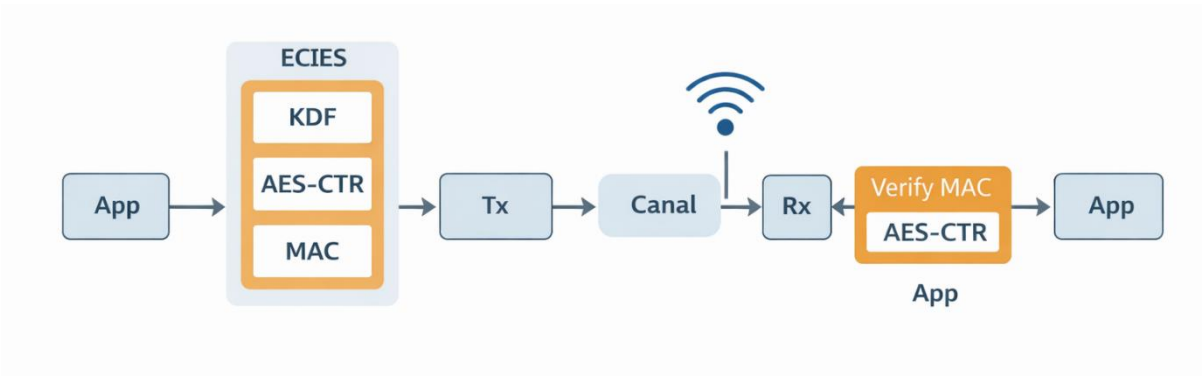
Bajo este enfoque, el flujo de datos se interpreta como una secuencia de transformación del mensaje:

1. Aplicación: genera el payload (datos sensados o mensaje de control).
2. Seguridad (ECIES): deriva el secreto compartido mediante ECDH a partir de material criptográfico preconfigurado/derivado determinísticamente; la variabilidad por paquete se introduce mediante IV aleatorio en AES-CTR.
3. Encapsulación y transmisión: el mensaje protegido se transmite por el canal acústico con los parámetros del escenario.
4. Recepción: el nodo receptor procesa el paquete, verifica la firma ECDSA y, solo si la verificación es correcta, descifra el payload.

Esta ubicación del mecanismo de seguridad se justifica porque permite evaluar el costo criptográfico como overhead en tamaño y tiempo de proceso, sin alterar la lógica del canal. En consecuencia, el canal y la pila inferior permanecen comparables entre escenarios, y la diferencia observada en métricas puede asociarse principalmente a la inclusión de seguridad y su efecto sobre el tamaño del mensaje, colas y temporización. El intercambio lógico de mensajes y la secuencia cifrar–enviar–verificar–descifrar se resume en la **Figura 23**.

Figura 23

Modelo de nodo y el punto de inserción del bloque de seguridad dentro del flujo de aplicación.



3.4.3.1 Flujo Tx/Rx con seguridad como transformación del mensaje

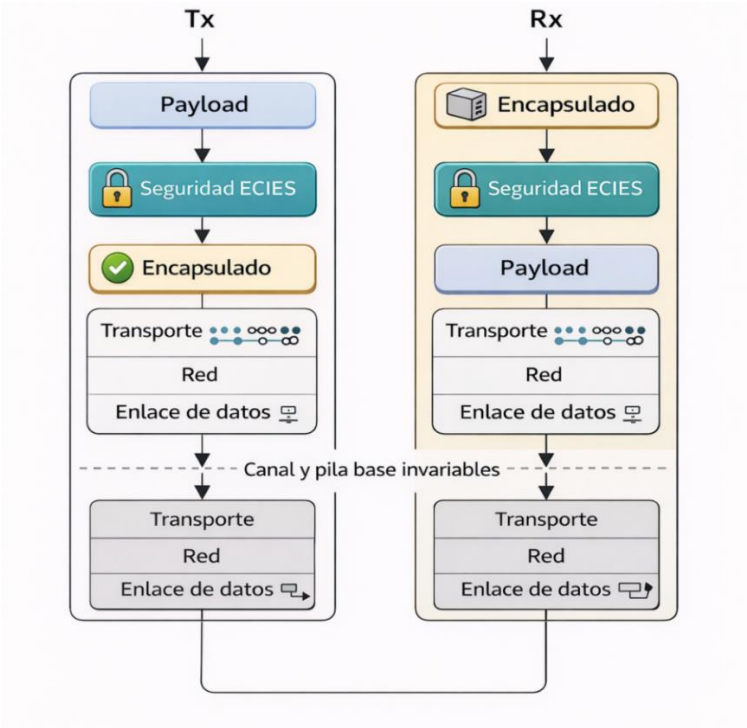
En el entorno simulado, la incorporación de seguridad se realiza a nivel de aplicación, preservando sin cambios el modelo base de canal acústico y la pila mínima de comunicación. En consecuencia, la red mantiene el mismo comportamiento subyacente en cuanto a propagación, retardo, pérdidas y acceso al medio; la diferencia entre perfiles se introduce exclusivamente como una transformación del mensaje previa a la transmisión y posterior a la recepción. Bajo esta interpretación, el payload generado por la aplicación no se "rediseña" a nivel de canal, sino que se encapsula mediante un bloque criptográfico que convierte el dato en claro en un artefacto protegido (mensaje seguro), el cual es el que finalmente ingresa al proceso de envío.

Operativamente, el flujo Tx/Rx se entiende como una cadena: (i) la aplicación produce datos (control o sensado), (ii) el bloque ECIES encapsula y protege el contenido, (iii) el paquete se transmite sobre el canal acústico manteniendo la configuración del escenario, y (iv) el receptor ejecuta la operación inversa para recuperar el payload únicamente si el mensaje cumple la política de aceptación definida. En la .

Figura 24 se puede observar cómo esta decisión de diseño permite atribuir, de forma controlada, el impacto observado en el desempeño a los efectos directos de la seguridad (sobrecarga y cómputo), sin introducir variaciones no controladas en capas inferiores.

Figura 24

Diagrama del nodo y punto de inserción del bloque de seguridad ECIES (AES-CTR + ECDSA) en el flujo Tx/Rx



La seguridad actúa como transformación del payload a nivel de aplicación: encapsulado antes de Tx y recuperación después de Rx, manteniendo invariante el canal y la pila base.

3.4.3.2 Política operativa de aceptación: verify → decrypt

El receptor adopta una política de aceptación verify → decrypt, alineada con el modelo de amenazas del entorno subacuático y con la necesidad de proteger recursos (tiempo de CPU y energía) frente a tráfico inválido o malicioso. En términos operativos, ante un paquete marcado como seguro, el nodo receptor valida primero autenticidad e integridad mediante la firma digital; únicamente si la verificación resulta satisfactoria procede a reconstruir el material de clave de sesión y ejecutar el descifrado.

Esta secuencia tiene dos implicaciones metodológicas. Primero, reduce la probabilidad de que un nodo invierta procesamiento adicional en mensajes alterados, inyectados o suplantados, lo que resulta coherente con amenazas típicas de canal y con la

evaluación de robustez. Segundo, establece un criterio determinista de descarte: los mensajes que no superan verificación se eliminan del flujo de procesamiento y no contribuyen a la carga del descifrado, estabilizando el comportamiento ante intentos de manipulación y facilitando la trazabilidad de eventos de rechazo (drops por verificación).

3.4.3.3 Implicaciones directas para la medición

En el diseño experimental adoptado, el canal acústico se mantiene como un componente invariante del escenario y se modela de forma explícita mediante UanSoundMedium, mientras que la seguridad se introduce únicamente como una transformación del mensaje en la capa de aplicación. Esta separación es central para el control experimental: cualquier variación observada entre el perfil base (NS) y el perfil seguro (S) no proviene de cambios en propagación, retardo o pérdidas del medio, sino del efecto directo del encapsulado criptográfico (campos añadidos y procesamiento) sobre la carga transmitida y el flujo operativo de aceptación. En otras palabras, el canal determina las condiciones físicas de la comunicación y permanece constante en todas las ejecuciones comparables; el bloque ECIES modifica la representación del payload antes del envío y condiciona su recuperación en la recepción mediante la política verify → decrypt. Con ello, el análisis posterior puede atribuir el impacto medido en desempeño a la seguridad, sin introducir sesgos por alteraciones del modelo de canal.

Al integrarse como transformación de la carga útil, la seguridad impacta de forma directa dos dimensiones medibles: (i) tamaño del mensaje, debido a campos criptográficos adicionales, y (ii) tiempo de procesamiento, por las operaciones de derivación de claves, cifrado/descifrado y firma/verificación. Estas variaciones se reflejan en métricas de red (p. ej., latencia extremo a extremo, throughput/goodput, PDR/PLR) y en métricas internas del bloque criptográfico (tiempos de cifrado/descifrado y tasa de verificación exitosa).

Bajo un enfoque experimental, esto permite atribuir el cambio de desempeño entre perfiles NS y S a una causa controlada: el costo incremental de encapsular seguridad sin alterar el canal. La instrumentación descrita en las secciones de medición y salida de resultados registra tanto los efectos en la comunicación como los eventos internos de aceptación/descarte, permitiendo que el Capítulo 4 cuantifique el impacto relativo de incorporar ECIES en distintos niveles de densidad y carga.

Con el punto de inserción definido a nivel conceptual y su relación directa con la medición, la siguiente etapa consiste en describir cómo este diseño se materializa en el software del proyecto: módulos de aplicación, estructura del paquete y la secuencia implementada de encapsulado seguro y validación en la recepción. Esta correspondencia diseño–implementación se formaliza en la Sección 3.7, culminando en la integración del bloque criptográfico dentro del flujo operativo de los nodos.

3.4.4 Modelo del canal acústico y supuestos de propagación

A diferencia de entornos terrestres, el canal submarino se caracteriza por una propagación más lenta, pérdidas dependientes de distancia y frecuencia, y variaciones por condiciones del medio. En el presente trabajo, el canal acústico se representa mediante un modelo que incorpora: (i) retardo de propagación, (ii) atenuación/pérdida de trayecto, y (iii) un componente de degradación que se manifiesta como errores o pérdidas bajo ciertas condiciones del enlace.

El retardo de propagación se determina de forma consistente con la velocidad del sonido asumida en el medio, de manera que el tiempo de tránsito del mensaje se vuelve función directa de la distancia entre nodos. Para la pérdida de trayecto, se adopta un enfoque donde la potencia recibida se reduce con la distancia y la frecuencia, representando el comportamiento típico de enlaces acústicos. Estos supuestos permiten que el canal sea

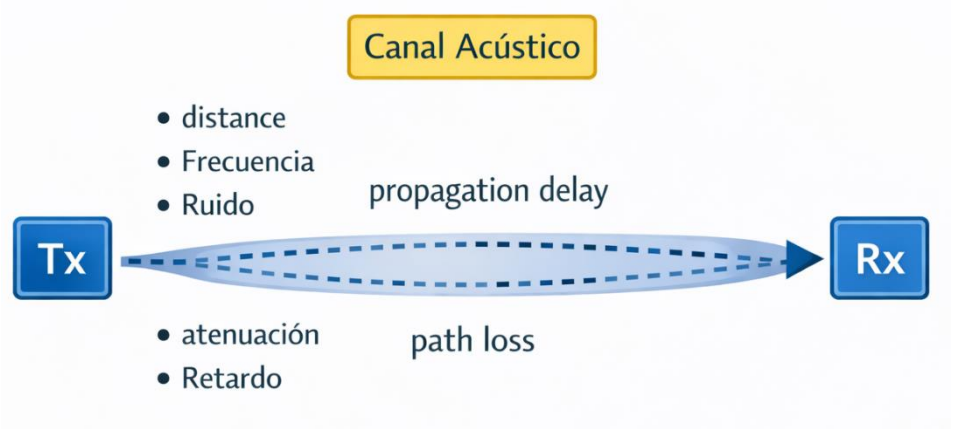
controlable experimentalmente, es decir, que el investigador pueda fijar condiciones (distancia, parámetros físicos y tráfico) para observar su efecto sobre el desempeño.

En el contexto de esta tesis, este modelado se utiliza con un propósito específico: asegurar que el canal y sus parámetros se mantengan constantes entre escenarios comparativos, de modo que el análisis del impacto de ECIES se realice en igualdad de condiciones. En otras palabras, el canal actúa como "banco de pruebas" estable donde se introduce la variación de seguridad como factor experimental principal.

La **Figura 25** presenta el esquema del canal acústico considerado y las variables principales que intervienen en la propagación.

Figura 25

Esquema del canal acústico considerado y las variables principales de propagación.



En esta sección se presenta una descripción sintética del modelo de canal acústico empleado, manteniendo la formulación matemática y sus parámetros detallados en la Sección 2.1.4. Con ello se evita redundancia y se preserva la trazabilidad de las ecuaciones base utilizadas en la simulación.

3.4.5 Configuración de tráfico y condiciones de ejecución

Para evaluar el desempeño de la red en condiciones comparables, el tráfico se define de forma parametrizable: tamaño de mensaje, intervalo de envío y duración de la simulación. En escenarios base, el tráfico representa una transmisión periódica; en escenarios con múltiples nodos, el tráfico se ajusta para representar carga agregada sobre el canal y observar el efecto combinado de contención, colas y retransmisiones.

A nivel experimental, cada escenario se ejecuta bajo perfiles diseñados para habilitar comparaciones directas:

- NS (No Security): transmisión sin protección criptográfica.
- S (Security): transmisión con ECIES (AES-CTR + ECDSA).
- A (Attack): transmisión con presencia de atacante, manteniendo constantes el canal y el tráfico.
- L (Listener): transmisión con presencia de atacante pasivo o en modo escucha que actúa como MITM.

La **Tabla 26** resume los escenarios experimentales (6 condiciones) y los parámetros de ejecución que se mantienen constantes para asegurar comparabilidad.

Tabla 26

Perfiles experimentales (Escenarios de estudio) y condiciones de ejecución (OMNeT++ 6.1)

Escenario	**Configuración**	**Factor que varía**	**Constantes (control)**	**Objetivo**
NS_Normal	Sin seguridad / sin atacante	Payload en claro (sin overhead)	N, canal acústico, tráfico base, despliegue, duración,	Línea base de desempeño/con ectividad

			semillas/repet., instrumentación	
NS_Atacante	Sin seguridad / atacante activo	Tráfico del atacante (más colisiones/colas)	*(igual)*	Medir degradación por ataque sin seguridad
NS_Listener	Sin seguridad / atacante pasivo	Escucha del canal (sin inyección)	*(igual)*	Evidenciar exposición del payload en claro
S_Normal	Seguridad E2E (ECDH+KDF+ AES-CTR+ECDSA) / sin atacante	Encapsulado + costo de cómputo	*(igual)*	Impacto de seguridad en delay/PDR/eficiencia
S_Atacante	Seguridad / atacante activo	Seguridad + perturbación; descartes por verificación	*(igual)*	Robustez (verify→decrypt) y trade-off bajo ataque
S_Listener	Seguridad / atacante pasivo	Seguridad bajo observación	*(igual)*	Confidencialidad: listener no recupera payload

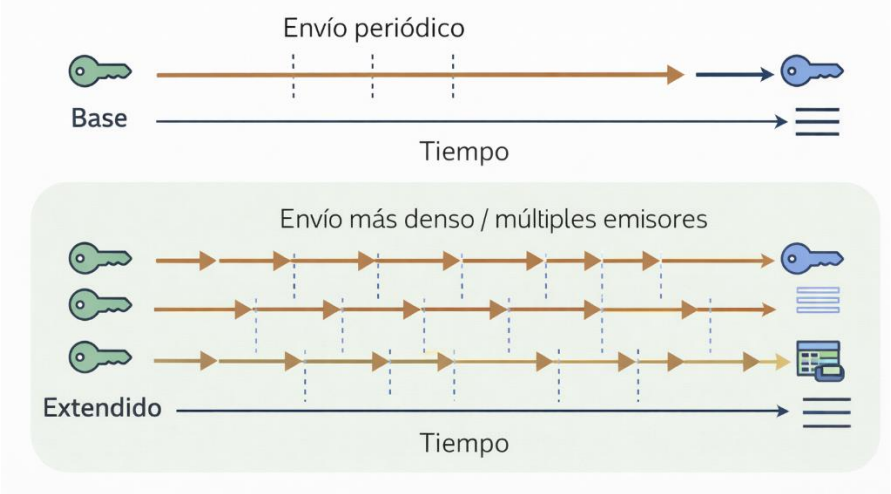
En todas las comparaciones NS vs S, se mantuvieron constantes el modelo de canal acústico, el patrón de tráfico y la topología del escenario Ei. Por tanto, cualquier variación

observada en retardo, PDR/PLR, throughput/goodput u overhead se atribuye al costo incremental del encapsulado criptográfico y su interacción con la contención del medio.

La **Figura 26** resume el patrón de tráfico considerado en escenarios base y cómo se incrementa la carga en escenarios extendidos.

Figura 26

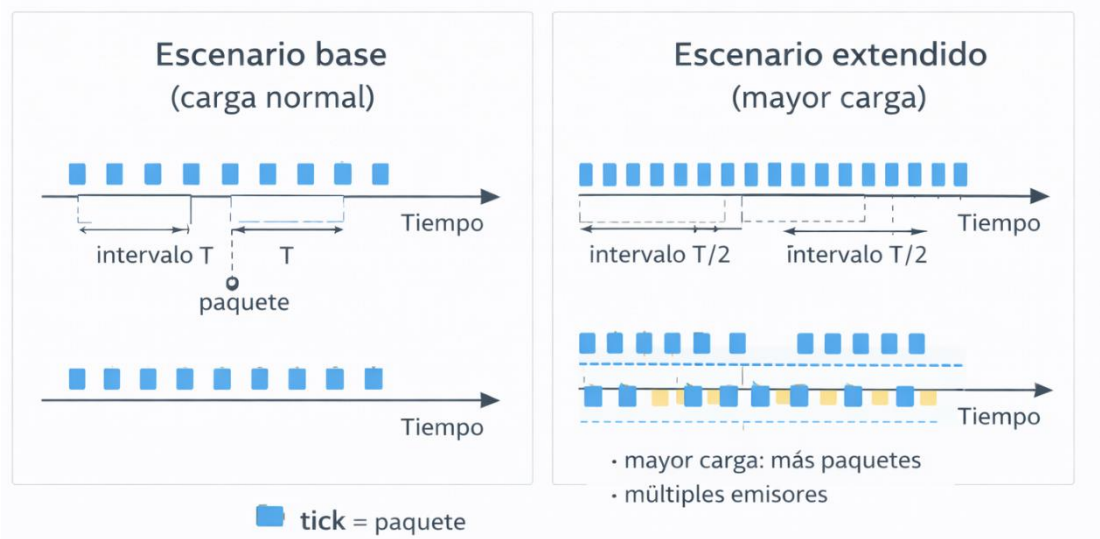
Patrón de tráfico considerado en escenarios base y en escenarios extendidos



La **Figura 27** resume el patrón de tráfico considerado en los escenarios base y cómo se incrementa la carga en escenarios extendidos.

Figura 27

Patrón de tráfico considerado en los escenarios base y cómo se incrementa la carga en escenarios extendidos.



3.4.6 Instrumentación y salida de resultados

La recolección de resultados se realiza mediante los mecanismos estándar de OMNeT++ para registrar métricas a lo largo del tiempo (vectores) y resúmenes por corrida (escalares). Con el fin de garantizar trazabilidad, cada escenario define explícitamente: parámetros de canal, parámetros de tráfico y configuración de ejecución. En la fase de resultados, estas salidas se consolidan para construir tablas y gráficas comparativas por escenario, distinguiendo ejecuciones sin seguridad frente a ejecuciones con ECIES (AES-CTR + ECDSA).

Para mantener reproducibilidad, la simulación se ejecuta con control de semillas y repeticiones, de manera que los resultados reportados correspondan a tendencias estables y no a ejecuciones aisladas. Este procedimiento se detalla posteriormente en la sección de métricas y en el capítulo de resultados, donde se presentan las comparaciones cuantitativas. La **Figura 28** muestra el flujo de instrumentación y consolidación de resultados desde la simulación hasta las tablas y gráficas de evaluación.

Figura 28

Flujo de instrumentación y consolidación de resultados desde OMNeT++

Con el entorno descrito, se habilita la ejecución controlada de escenarios de construcción (S0–S3) y una campaña de evaluación por densidad (N = 20, 50 y 100), detallada en la Sección 3.5, bajo seis escenarios finales (NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener) y el alcance del atacante.

3.5 Diseño de escenarios experimentales

El objetivo de esta sección es definir los escenarios de simulación que permiten (i) construir y verificar el funcionamiento del modelo de comunicación y del mecanismo de seguridad, y (ii) evaluar el desempeño bajo incrementos de densidad de red. Para mantener comparabilidad entre ejecuciones, cada escenario se ejecuta bajo condiciones controladas del canal y del tráfico, variando únicamente los factores de interés: densidad (número de nodos), perfil de seguridad (sin/con ECIES) y presencia de atacante cuando aplique.

Con el fin de mantener trazabilidad entre la implementación (nombres ya existentes en OMNeT++) y la nomenclatura formal de esta tesis, se adopta una convención dual: S0–S3 para escenarios de construcción/verificación y D20–D100 para la evaluación por densidad (N = 20, 50 y 100). Esta convención evita confusión entre “escenarios de desarrollo” y “escenarios de medición”.

3.5.1 Convención de escenarios y equivalencias

En la implementación, los escenarios se mantienen con su denominación original dentro del proyecto de simulación (por ejemplo, Base/T2/T3/T4). En esta tesis, dichos

escenarios se referencian como S0–S3 (construcción) y D20–D100 (evaluación por densidad), conforme a la equivalencia presentada en la **Tabla 27**.

La equivalencia entre nomenclatura de implementación (OMNeT++) y nomenclatura académica (S0–S3, D20–D100).

Tabla 27

Equivalencias de escenarios (implementación ↔ tesis)

Nombre en OMNeT++	**Tipo**	**Propósito metodológico**	**Relación con evaluación**
Base (conectividad)	Construcción	Verificar conectividad y estabilidad del modelo base sin seguridad.	Equivale conceptualmente a NS_Normal en topología mínima (2 nodos) para validación.
T2 / Escenario base	Construcción	Verificar el flujo funcional de seguridad extremo a extremo (ECIES + AES-CTR + ECDSA).	Equivale conceptualmente a S_Normal en topología mínima (2 nodos) para validación del flujo seguro.

T3 (verificación medio/perfiles)	Construcción	Verificar el medio acústico compartido (UanSoundMedium) y el comportamiento half-duplex de la NIC UAN; validar la conmutación de perfiles NS/S y de los modos Normal/Atacante/Listener por parametrización (t3.ini).	Paso de verificación previo a la campaña final: confirma canal común y selección de perfiles/amenazas sin introducir el barrido de densidad ni la comparación de métricas finales.
T4 (plantilla escalable) / Campaña final	Plantilla final	Plantilla parametrizable para escalar densidad (D20, D50 y D100) y conmutar los seis escenarios finales (NS/S en modos Normal/Atacante/Listener) mediante parámetros en t4.ini.	Habilita el barrido por densidad D20–D100 (N={20,50,100}) y la ejecución comparable de los seis escenarios para obtener las métricas del Capítulo 4.

3.5.2 Escenarios de construcción y verificación incremental (S0–S3)

Los escenarios Base y T2–T4 se emplean como ruta incremental para disminuir riesgo experimental. En términos metodológicos, estos escenarios validan que el entorno y el flujo de comunicación son correctos antes de ejecutar la campaña final de evaluación por densidad (D20, D50 y D100; N = 20, 50 y 100).

En esta tesis se adopta la siguiente correspondencia funcional: (i) T2 (S1): verificación del mecanismo ECIES en topología mínima (2 nodos); (ii) T3 (S2): verificación del medio acústico (UanSoundMedium/half-duplex) y de la conmutación de perfiles NS/S y modos de atacante por parametrización; y (iii) T4 (S3): plantilla escalable utilizada en la campaña final (6 escenarios × 3 densidades).

3.5.2.1 Escenario base de conectividad

El escenario base corresponde a una topología mínima orientada a comprobar que los nodos transmiten y reciben sobre el canal acústico bajo una configuración estable. En este escenario no se introduce seguridad; su finalidad es establecer un baseline funcional y confirmar que la instrumentación puede registrar eventos de envío/recepción y temporización.

Criterio de aceptación: comunicación estable sin errores de ejecución y evidencia de recepción conforme al patrón de tráfico definido.

Este incremento corresponde al primer entregable del flujo Kanban, donde se valida la conectividad y la estabilidad del modelo base antes de introducir mecanismos de seguridad

3.5.2.2 S1 (T2) – Escenario de verificación del mecanismo ECIES (2 nodos)

S1 introduce la protección extremo a extremo en la comunicación, habilitando el flujo lógico de seguridad en el emisor y el proceso de verificación/recuperación en el receptor. El propósito es comprobar el comportamiento funcional del esquema: cifrado con AES-CTR, generación/ verificar la firma ECDSA y descifrado únicamente tras verificación satisfactoria.

Criterio de aceptación: el receptor acepta y entrega el payload solo cuando el firma ECDSA es válido; ante alteración o inconsistencia, el mensaje se descarta (si el ataque/alteración está habilitado en esa prueba). Este incremento se gestiona como una

unidad Kanban independiente, dado que definición de realizado exige verificación funcional del flujo ECIES (AES-CTR + ECDSA) sin modificar el canal ni la topología

3.5.2.3 S2 (T3) – Verificación del medio y conmutación de perfiles

S2 corresponde al escenario T3 dentro de la implementación y se utiliza como un paso intermedio de verificación antes de escalar la red. Su propósito es comprobar que el medio acústico compartido (UanSoundMedium) está correctamente instanciado a nivel de red, que la interfaz UAN opera en modo half-duplex y que el flujo JOIN/JOIN_REPLY es consistente bajo el stack base.

Adicionalmente, en este escenario se valida la conmutación de perfiles y amenazas por parametrización, de manera que los modos Normal/Atacante/Listener y el perfil NS/S se activen sin modificar el modelo base ni el código. La Figura 20 documenta evidencia del canal común y la operación half-duplex en T3, mientras que la Figura 43 muestra extractos representativos de la selección de perfiles en el archivo de configuración t3.ini.

Criterio de aceptación: ejecución estable del flujo de incorporación (JOIN) y tráfico de datos observable; activación reproducible de flags de perfil (p. ej., cifrar) y de modo del atacante (activo/pasivo) conforme a la configuración; y evidencia de que el sistema conserva la misma pila base de canal y NIC entre ejecuciones comparables.

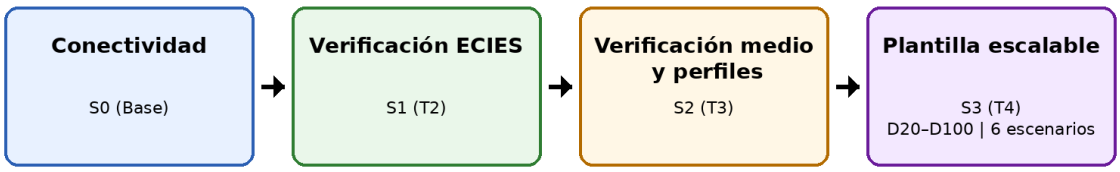
3.5.2.4 S3 (T4) – Plantilla escalable y conmutación de escenarios (6 condiciones)

S3 (T4) define una plantilla de escenario orientada a escalado y comparación. La red se parametriza para variar la densidad (D20, D50 y D100) y conmutar los seis escenarios finales (NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener) mediante parámetros en la campaña t4.ini.

Criterio de aceptación: ejecución estable al incrementar densidad hasta el máximo evaluado (D100 / N = 100) y conmutación reproducible entre los seis escenarios (NS/S en modos Normal/Atacante/Listener) sin modificar el modelo base como se ve en la **Figura 29**.

Figura 29

La ruta incremental de construcción S0 → S1 → S2 → S3



Este incremento cierra la fase de construcción incremental gestionada con Kanban, al consolidar una plantilla parametrizable (T4) que habilita la campaña final por densidad (D20–D100) y sus seis escenarios comparables.

3.5.3 Barrido de densidad de red (N = 20, 50, 100)

La campaña principal de evaluación se define por el número de nodos participantes. La hipótesis metodológica es que el incremento de densidad aumenta contención, colas y efectos agregados del canal; por tanto, permite observar con mayor claridad el trade-off entre seguridad y desempeño en UWSN.

Se consideran tres densidades representativas:

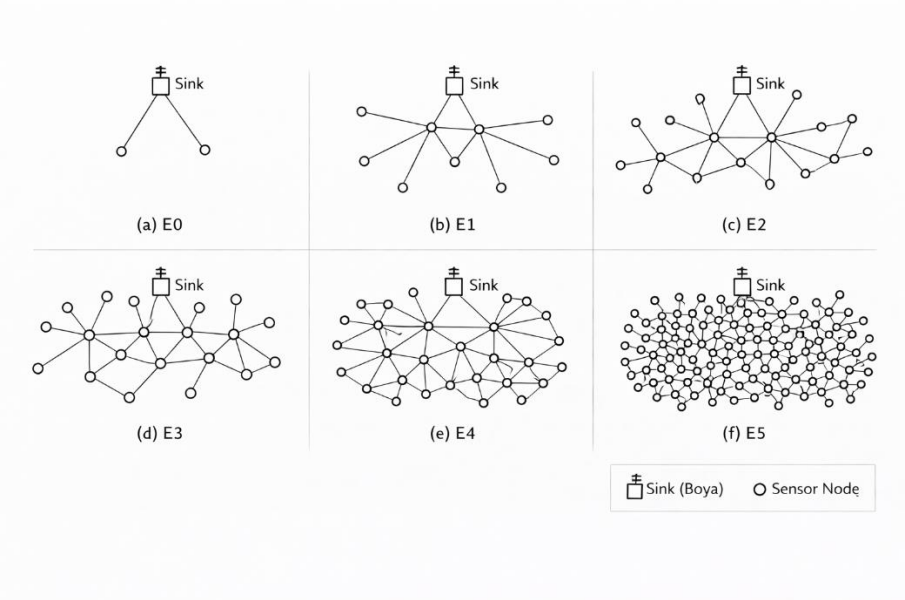
- D20: 20 nodos (carga moderada).
- D50: 50 nodos (carga alta).
- D100: 100 nodos (estrés de red).

En todos los casos, la comparación se realiza ejecutando los seis escenarios definidos (NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener) para cada

densidad N, manteniendo constantes el canal y el tráfico base para garantizar comparabilidad entre corridas como se mira en la **Figura 30**.

Figura 30

Las topologías asociadas a N = 20, 50 y 100 evidenciando el incremento progresivo de densidad.



3.5.4 Escenarios de seguridad y amenaza (6 condiciones)

Con el fin de aislar el efecto del mecanismo criptográfico y del atacante, se define un diseño factorial 2×3: seguridad {NS, S} y amenaza {Normal, Atacante activo, Listener pasivo}. Las seis condiciones resultantes son:

• NS_Normal: sin seguridad, sin atacante (línea base).

• NS_Atacante: sin seguridad, con atacante activo (Atackapp) que inyecta/perturba tráfico.

• S_Normal: con seguridad (esquema híbrido basado en ECDH + KDF + AES-CTR + firma ECDSA), sin atacante.

• S_Atacante: con seguridad, con atacante activo (Atackapp).

• NS_Listener: sin seguridad, con listener pasivo (escucha del medio sin inyección).

• S_Listener: con seguridad, con listener pasivo (escucha del medio).

- Este diseño permite comparaciones directas dentro de una misma densidad N: cualquier diferencia en métricas se atribuye a (i) overhead y procesamiento por seguridad (NS vs S) y/o (ii) degradación por amenaza (Normal vs Atacante vs Listener), evitando sesgos por cambios de topología o modelo de canal.

3.5.5 Escenario con atacante y alcance del modelo de amenaza

El atacante se modela como un nodo adicional integrado en la red y se activa mediante parámetros de configuración, dando lugar a los escenarios NS_Atacante/S_Atacante (atacante activo) y NS_Listener/S_Listener (atacante pasivo). Metodológicamente, el atacante permite evaluar dos aspectos: (i) el impacto sobre desempeño (por contención/colas o por tráfico adicional si aplica) y (ii) el comportamiento del esquema de seguridad ante observación o manipulación.

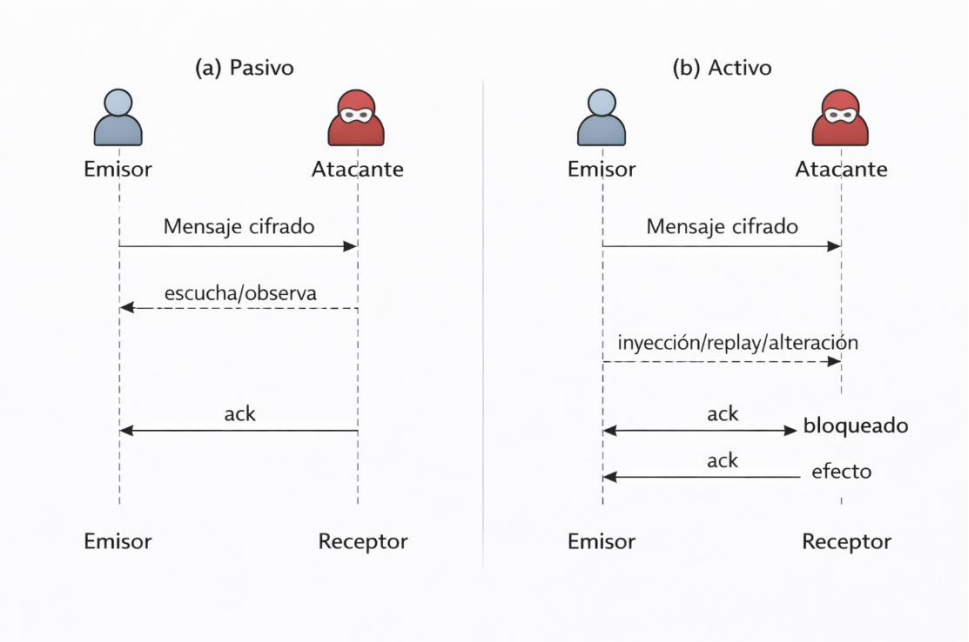
Se consideran dos modos de operación del atacante:

- Atacante pasivo: observa el canal y recolecta tráfico sin inyectar ni alterar mensajes. Este modo se utiliza para discutir confidencialidad, dado que la observación no debe permitir recuperar el payload cuando la seguridad está habilitada.
- Atacante activo: intenta alterar, reinyectar o perturbar la comunicación (según el alcance implementado). Este modo permite discutir integridad/autenticación, dado que el receptor debe rechazar mensajes con firma ECDSA inválida en el perfil con seguridad.

En caso de que el alcance implementado esté limitado a observación pasiva o a un subconjunto de acciones, esto se declara explícitamente como parte del modelo de amenaza aplicado. En la **Figura 31** se puede observar la interacción que se desea demostrar.

Figura 31

La interacción emisor–receptor con la inclusión del atacante.



3.5.6 Parámetros controlados y trazabilidad hacia métricas

Para asegurar validez experimental, se adoptan dos grupos de variables:

- Variables controladas (constantes): configuración del canal acústico, patrón de tráfico base, duración de simulación, instrumentación y método de recolección, así como la configuración reproducible (semillas y repeticiones) definida en la sección 3.6 Métricas, recolección y procesamiento de resultados.
- Variables manipuladas: densidad de red (N = 20, 50, 100) y escenario (6 condiciones: NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener).

La relación entre escenarios y métricas se consolida en la Sección 3.6 Métricas, recolección y procesamiento de resultados, donde se formaliza el cálculo operacional de delay E2E, throughput/goodput, PDR, overhead criptográfico y demás indicadores. La **Tabla 28** resume la trazabilidad de los escenarios propuestos.

Tabla 28

La trazabilidad entre escenarios, variables manipuladas y métricas

Escenario	Propósito	Variables manipuladas	Perfil(es)	Métricas objetivo
S0 (Base)	Verificar conectividad y baseline funcional.	Topología mínima (2 nodos).	**NS**	Conectividad, delay E2E, PDR (verificación).
S1 (T2)	Verificar flujo funcional de seguridad.	Seguridad activada.	**S**	Overhead (bytes), delay E2E (impacto inicial), aceptación/descartes (firma ECDSA).
S2 (T3)	Verificar el medio acústico y la conmutación de perfiles/amenazas.	Activación de NS/S y modos Normal/Atacante/Listener por parámetros (t3.ini).	NS/S × Normal/Atacante/Listener (red de verificación).	Evidencia de canal común/half-duplex, flujo JOIN/JOIN_REPLY y consistencia de instrumentación.

S3 (T4)	Plantilla escalable para la campaña final por densidad.	Densidad parametrizable (D20–D100) + conmutación de 6 escenarios.	6 escenarios (NS/S × Normal/Atacante/Listener)	Estabilidad al escalar, consistencia de medición y generación de tablas/gráficas.
D20 (20)	Carga moderada.	#Nodos=20	6 escenarios	Delay E2E, throughput/goodput, PDR/PLR, overhead.
D50 (50)	Carga alta.	#Nodos=50	6 escenarios	Delay E2E, throughput/goodput, PDR/PLR, overhead.
D100 (100)	Estrés de red.	#Nodos=100	6 escenarios	Delay E2E, throughput/goodput, PDR/PLR, overhead; robustez ante atacante (cuando aplica).

Nota: Variables controladas en todas las comparaciones: modelo de canal acústico, patrón de tráfico base, parámetros físicos del despliegue, duración de corrida, configuración de semillas/repeticiones e instrumentación (scalars/vectors).

Perfiles/escenarios: NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener (equivalente a NS/S en modos Normal/Atacante/Listener).

3.6 Métricas, recolección y procesamiento de resultados

Esta sección formaliza las métricas empleadas para evaluar el desempeño de la red bajo los escenarios definidos en la Sección 3.5 Diseño de escenarios experimentales. El objetivo es establecer definiciones operacionales (qué se mide, dónde se mide y en qué unidades) y asegurar que la recolección sea trazable y comparable entre los seis escenarios de evaluación (NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener). En consecuencia, se priorizan métricas que reflejan tanto la calidad del enlace como el costo asociado al mecanismo de seguridad en condiciones de densidad creciente y bajo amenaza.

3.6.1 Métricas de evaluación y definiciones operacionales

Las métricas seleccionadas se agrupan en: (i) desempeño extremo a extremo, (ii) confiabilidad de entrega, (iii) eficiencia de transmisión y (iv) costo/overhead por seguridad. Adicionalmente, cuando se ejecuta el perfil con atacante, se considera una métrica funcional relacionada con el comportamiento de integridad/autenticación (por ejemplo, rechazo de mensajes inválidos).

La **Tabla 29** resume las métricas principales, su definición operacional y el punto lógico de medición.

Tabla 29

Métricas, definición operacional y puntos de medición

Métrica	Símbolo	Definición operacional (qué representa)	Punto lógico de medición	Unidad	Tipo de salida
Retardo extremo a extremo	D_{e2e}	Diferencia entre el instante de generación del mensaje en el emisor y el instante en que el payload es entregado a la aplicación en el receptor.	Emisor (generación) / Receptor (entrega)	s o ms	Vector + scalar (promedio)
Jitter (variabilidad del retardo)	(J)	Dispersión del retardo E2E. En esta tesis se reporta como desviación estándar (σ) del Delay_E2E y se complementa con percentiles p5/p95 cuando corresponde.	Receptor (gateway)	s o ms	Scalar (DE + percentiles)
Throughput	(T)	Tasa de bits/bytes recibidos en el receptor por unidad de tiempo, considerando el tamaño total del mensaje transportado (payload + overhead).	Gateway (sumidero)	bps o B/s	Scalar
Goodput	(G)	Tasa efectiva de payload útil entregado por unidad de tiempo (excluye overhead	Gateway (payload entregado)	bps o B/s	Scalar

		criptográfico y campos adicionales).			
Tasa de entrega de paquetes	(PDR)	Cociente entre número de paquetes recibidos válidos y número de paquetes transmitidos.	Emisor (tx) / Receptor (rx válido)	%	Scalar
Pérdida de paquetes	(PLR)	Complemento de (PDR): paquetes no entregados respecto a transmitidos.	Emisor/ Receptor	%	Scalar
Overhead por seguridad	(OH)	Incremento de tamaño debido a seguridad: diferencia entre el tamaño del mensaje seguro y el mensaje sin seguridad (absoluto y/o relativo).	Enlace lógico de la aplicación	B o %	Scalar
Rechazo por integridad/autenticación (si hay ataque activo)		Proporción de mensajes descartados por fallo de verificación ECDSA respecto a mensajes recibidos.	Receptor (verificación)	%	Scalar

Nota: En el perfil E_i-S, la diferencia entre throughput y goodput refleja el costo del overhead criptográfico (campos adicionales como R, IVy firma), mientras que el efecto en D_{e2e} captura la interacción entre mayor tamaño de trama, colas/contención y temporización del canal. En el perfil E_i-A, la métrica R_{sig} solo aplica si el atacante induce alteración/reinyección; si el atacante es estrictamente pasivo, se omite R_{sig}y el análisis se centra en desempeño y confidencialidad.

Definiciones operacionales:

(1) Retardo extremo a extremo: Delay_E2E = t_entrega - t_generación.

(1a) Jitter: se define como la variabilidad del retardo E2E y se cuantifica mediante la desviación estándar (σ) de Delay_E2E en cada corrida; adicionalmente se reportan percentiles p5 y p95 para caracterizar dispersión y colas.

(2) PDR = (Paquetes válidos recibidos) / (Paquetes transmitidos).

(3) PLR = 1 - PDR.

(4) Overhead absoluto: OH = size_S - size_NS.

(5) Overhead relativo: OH% = (OH / size_NS) × 100.

(6) Throughput considera el tamaño total transportado (payload + overhead), mientras que Goodput considera únicamente el payload útil entregado. Para fines de evaluación, en esta tesis throughput y goodput se reportan exclusivamente en el gateway (sumidero) como punto común de recepción.

3.6.2 Control experimental: variables controladas y variables manipuladas

Para garantizar validez experimental, el diseño controla el entorno y modifica únicamente los factores definidos en el plan de escenarios. En particular, las comparaciones "sin seguridad" vs "con seguridad" se realizan manteniendo constantes el canal y el patrón de tráfico base, de modo que el efecto observado se asocie al uso de ECIES (AES-CTR + ECDSA) y su impacto indirecto en la carga del canal. Las variables controladas y manipuladas se resumen en la **Tabla 30**.

Tabla 30

Variables controladas vs variables manipuladas

Tipo	Variables	Descripción
Controladas (constantes)	Modelo del canal acústico, parámetros físicos del despliegue, patrón de tráfico	Se mantienen iguales dentro de una campaña

	base, duración de corrida, instrumentación, método de exportación de resultados	para permitir comparaciones directas.
Manipuladas (experimentales)	Densidad de red (D), perfil (NS/S/A), presencia/modo del atacante (si aplica)	Factores que se varían para observar su efecto en las métricas.

3.6.3 Repetibilidad y configuración de campañas (semillas y repeticiones)

La repetibilidad se asegura mediante la ejecución de escenarios bajo configuraciones deterministas y repetibles, en las que la variabilidad inherente a contención/colas se observa a través de múltiples corridas controladas por semillas. Para sostener comparaciones justas, se utiliza un conjunto consistente de semillas por densidad N y escenario, de modo que las comparaciones NS vs S (y Normal vs Atacante vs Listener) se realicen bajo condiciones estocásticas equivalentes cuando aplique.

En esta tesis, cada combinación (N, escenario) se ejecuta con n = 3 repeticiones, reportando en el Capítulo 4 medidas agregadas (media y dispersión) para evitar sesgo por ejecuciones particulares.

3.6.4 Consolidación de resultados y postprocesamiento

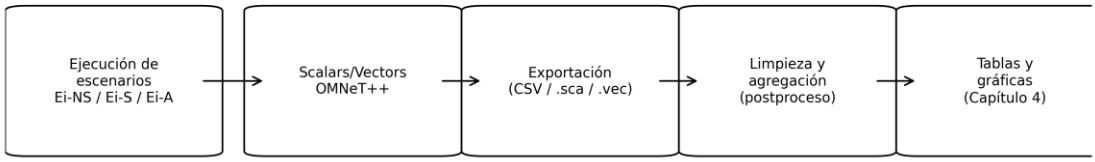
Los resultados recolectados se registran como series temporales (vectores) y resúmenes por corrida (escalares). El análisis posterior se enfoca en construir comparaciones por:

- Densidad: N = {20, 50, 100}.
- Escenario: NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener.
- Métrica: D_{e2e}, T, G, PDR/PLR, OH y R_{sig}

En el Capítulo 4 se reportan tablas y gráficas comparativas por métrica, destacando tendencias al aumentar la densidad y cuantificando el impacto relativo de incorporar seguridad. Para evitar sesgo por ejecuciones particulares, el reporte se basa en agregados por escenario/perfil (por ejemplo, media y desviación estándar) y, cuando sea pertinente, se discuten valores extremos (picos) en series temporales. La **Figura 32** muestra el flujo de datos desde la ejecución de escenarios hasta la generación de tablas y gráficas comparativas.

Figura 32

Flujo de datos desde la ejecución de escenarios hasta la generación de tablas y gráficas comparativas



Flujo de resultados: de la simulación a la evidencia cuantitativa

Con las métricas y el esquema de recolección definidos, el siguiente capítulo presenta la ejecución de la campaña N = {20, 50, 100} y el análisis comparativo entre los seis escenarios, cuantificando el impacto del mecanismo de seguridad (AES-CTR + firma ECDSA) y la presencia de atacantes activo/pasivo según corresponda.

3.6.5 Entrega válida, datagrama y eventos de descarte (drop)

En esta investigación, un mensaje se considera entregado cuando el paquete es recibido y aceptado por el receptor y el contenido es entregado a la aplicación (datagrama de aplicación). Bajo el perfil seguro (S), la entrega válida requiere además que el paquete supere la política operativa verify → decrypt; es decir, únicamente se contabiliza como entrega aquel DATA que, tras verificación exitosa, es descifrado y procesado por la aplicación del gateway.

En contraste, el término drop se utiliza para describir cualquier mensaje que no alcanza la entrega válida. En el análisis, este evento puede deberse a condiciones del canal (pérdida/interferencia), a saturación del sistema (colas y contención bajo carga o ataques tipo DoS), o a descarte explícito por seguridad (verificación fallida). Esta distinción es clave para interpretar correctamente las métricas del Capítulo 4, evitando atribuir al canal lo que corresponde a un rechazo criptográfico o a congestión inducida por ataque.

3.7. Configuración e implementación del proyecto en OMNeT++

Esta sección documenta los artefactos técnicos principales del proyecto (estructura, archivos NED/INI, definición de mensajes y módulos de código), con el objetivo de garantizar trazabilidad y reproducibilidad del escenario experimental.

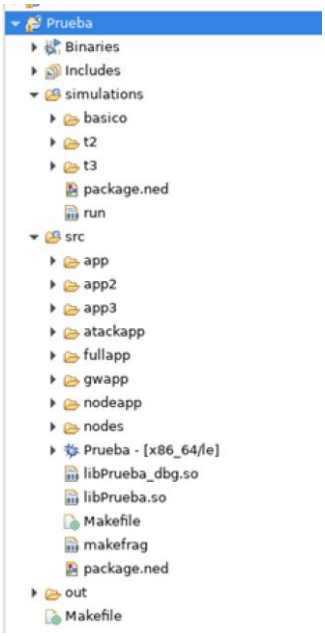
3.7.1 Estructura del proyecto y módulos principales

La

Figura 33 presenta la estructura general del proyecto. Las configuraciones de simulación se encuentran en el directorio "simulations/", mientras que los módulos C++ y definiciones de mensajes se ubican en "src/". Esta separación permite mantener escenarios reproducibles sin modificar el código base para cada perfil.

Figura 33

Estructura del proyecto (Prueba.zip) y localización de escenarios y módulos



Los módulos principales utilizados para la propuesta se resumen en la **Tabla 31**, destacando sus responsabilidades dentro del flujo sensores–gateway–atacante.

Tabla 31

Módulos principales del proyecto y responsabilidades

Componente	Archivo/Módulo	Responsabilidad
Sensor	src/fullapp/FullApp.cc	Genera/recibe mensajes; cifra/descifra y firma/verifica cuando cifrar=true.
Gateway	src/gwapp/gwapp.cc	Coordina JOIN/DATA; valida firma; descifra y gestiona lista de nodos.
Atacante	src/atackapp/atackapp.cc	Escucha (listener) y/o inyecta mensajes según perfil; evalúa robustez del sistema.
Nodo base	src/nodeapp/NodeApp.cc	Soporte de mensajes y comportamiento base en escenarios mínimos (T2).
Mensajes	src/nodeapp/AppNodePacket.msg	Define estructura del paquete: tipo, data, iv, publicKey, signature.

3.7.1.1 FullApp (nodo sensor): rol operativo y ciclo de comunicación

FullApp representa la lógica principal del nodo sensor en la simulación. Su rol operativo consiste en: (i) ejecutar el proceso de incorporación del nodo al dominio de comunicación (join/registro), (ii) generar tráfico de aplicación conforme al perfil experimental (frecuencia y carga), y (iii) procesar mensajes entrantes entregando el payload a la lógica de consumo. La activación de seguridad se define como condición de operación controlada por configuración: cuando el perfil seguro está habilitado, FullApp encapsula el contenido antes de transmitir; cuando está deshabilitado, el payload se envía en formato base, manteniendo invariante la lógica de tráfico para asegurar comparabilidad entre ejecuciones.

3.7.1.2 Gwapp (Gateway): coordinación y convergencia

Gwapp modela la lógica del gateway/boya como punto de convergencia. Su responsabilidad central es coordinar el establecimiento operativo con los sensores (intercambios de registro y control) y consolidar la recepción de datos. Desde el punto de vista experimental, el gateway sirve como referencia estable para observar el comportamiento de red bajo densidad creciente, y para centralizar la política de aceptación del perfil seguro. En modo S, Gwapp ejecuta verificación previa y solo en caso de éxito descifra y acepta el payload; en modo NS, procesa contenido en claro con el mismo flujo de control, contribuyendo a aislar el efecto criptográfico.

3.7.1.3 Atackapp (atacante activo): alcance y acciones modeladas

Atackapp formaliza un atacante activo incorporado en escenarios donde se requiere evaluar robustez frente a manipulación del canal. Su propósito no es vulnerar primitivas criptográficas, sino representar capacidades típicas de un atacante: inyección, alteración o perturbación de mensajes, y/o intentos de suplantación a nivel de tráfico. En coherencia con el alcance del trabajo, no se asume acceso a claves privadas legítimas ni capacidad de falsificar firmas; por tanto, el valor experimental del módulo se concentra en observar cómo

el sistema responde a tráfico adversarial y cómo la política verify → decrypt limita la aceptación de mensajes no confiables.

3.7.1.4 Listener (atacante pasivo): observación y objetivo

Listener representa un atacante pasivo cuyo objetivo es observar el tráfico en el medio sin intervenir en el contenido. Su función es registrar actividad de comunicación (ritmo de mensajes, patrones de intercambio y tamaño de paquetes). En escenarios NS, el observador pasivo ilustra el riesgo de exposición del payload en claro; en escenarios S, evidencia que, aun con visibilidad del tráfico y metadatos, el contenido permanece protegido por el encapsulado criptográfico. Esta separación entre atacante pasivo y activo favorece una evaluación incremental y consistente con el modelo de amenazas.

3.7.1.5 Relación Apps ↔ requerimientos ↔ métricas

La asignación de roles entre FullApp, Gwapp, Atackapp y Listener operacionaliza los requerimientos del sistema: protección extremo a extremo a nivel de aplicación, aceptación condicionada por verificación y comparabilidad experimental entre perfiles. Al mismo tiempo, habilita puntos consistentes de medición: generación/recepción de tráfico, eventos de validación y descarte, y efectos de sobrecarga sobre el desempeño de red. La

Tabla 32 sintetiza esta correspondencia entre aplicación, rol operativo, tipo de interacción, función experimental y métricas asociadas, asegurando coherencia entre la definición de componentes, el modelo de amenazas y los indicadores que se reportan posteriormente.

Tabla 32

Aplicación / Rol / Interacciones / Función experimental / Métricas asociadas.

Aplicación	**Rol**	**Interacciones principales**	**Función experimental**	**Métricas asociadas**
FullApp	Nodo sensor (emisor/receptor)	JOIN, envío DATA, recepción de control/respuesta	Generación de tráfico y encapsulado opcional de seguridad; punto primario de medición en el sensor	Latencia E2E y jitter (σ) en JOIN/DATA; PDR/PLR; tamaño de paquete; tiempos de cifrado/descifrado; bytes por mensaje. (Throughput y goodput se reportan en el gateway).
Gwapp	Gateway/Boya (convergencia)	JOIN_REPLY, recepción DATA, control (si aplica)	Punto de aceptación del sistema; referencia	Verificación exitosa/fallida; drops por

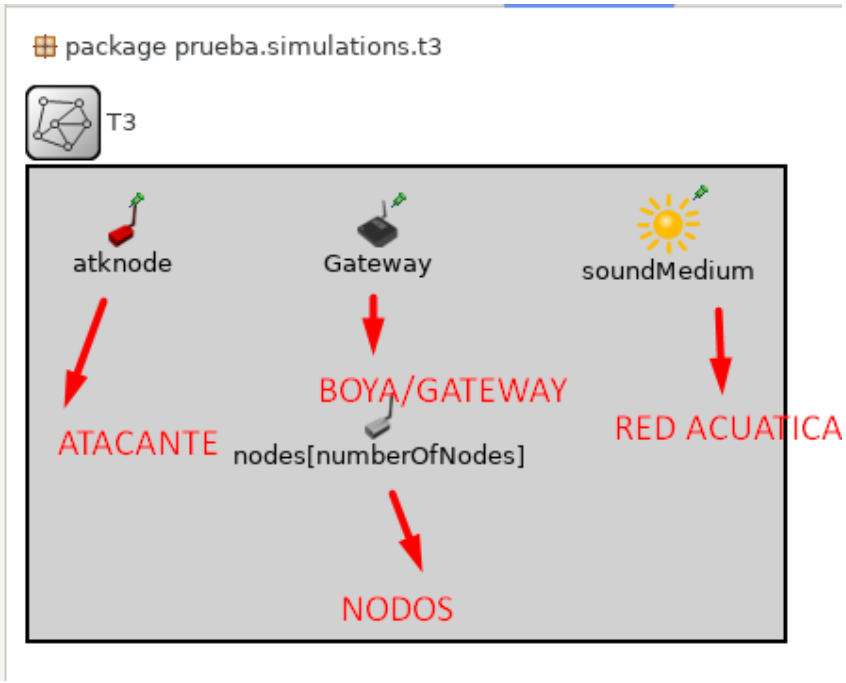
			estable para comparación	verificación; retardo y jitter; carga recibida; throughput/goodput; bytes agregados.
Atackapp	Atacante activo	Inyección/alteración/perturbación	Evaluar robustez sin suponer ruptura criptográfica	Rechazos por verificación; variación PDR/PLR; cambios en latencia/contención
Listener	Atacante pasivo	Observación del tráfico	Evidenciar exposición de actividad/tamaño vs confidencialidad	Tamaño observado; frecuencia/patrón; diferencia NS vs S en visibilidad del payload

Nota: seed-set afecta aleatoriedad del motor de simulación; no define identidad criptográfica. La identidad se pre-provisiona por configuración (gwpass/nodepass/useIndexAsKey), garantizando comparabilidad entre perfiles sin introducir tráfico adicional de distribución de claves.

En la **Figura 34** podemos ver la relación directa entre los módulos antes mencionados y su composición en el escenario del .NED.

Figura 34

Relación funcional y experimental entre los módulos de aplicación implementados en el escenario.



3.7.2 Definición del escenario y medio acústico (NED)

La definición de la red se realiza mediante archivos NED, donde se declaran nodos, conexiones lógicas y el medio acústico. Como se observa en la

Figura 35, el escenario referencia explícitamente un medio UAN (por ejemplo, UanSoundMedium), lo cual permite modelar la propagación y condiciones del canal subacuático dentro del simulador.

Figura 35

Ejemplo de definición NED con medio acústico (UanSoundMedium) y nodos del escenario

```
Graficas.anf  t3.ned  t3.ini  t2.ini  App2.cc  App2.h  t2.ned  red1.ned ×

1
2  package prueba.simulations.basico;
3
4  import Uan.physical.UanSoundMedium;
5  import prueba.nodes.Node;
6  import prueba.app.App;
7
8  network Red1
9  {
10     @display("bgb=386,252");
11     submodules:
12         soundMedium: UanSoundMedium {
13             @display("p=316,38;is=l");
14         }
15         node1: Node {
16             @display("p=115,130;is=l");
17         }
18         node2: Node {
19             @display("p=234,130;is=l");
20         }
21 }
22
```

3.7.3 Formato de mensajes de aplicación

El intercambio de información se modela a nivel de aplicación mediante un paquete dedicado. La **Figura 36** muestra la definición del tipo de mensaje (JOIN_REQUEST, JOIN_REPLY, DATA, TXCONFIG) y los campos necesarios para la operación con seguridad: "data" (payload o ciphertext), "iv" (vector de inicialización para CTR), "publicKey" (clave pública comprimida) y "signature" (firma ECDSA).

Figura 36

Definición del paquete AppNodePacket y campos utilizados por el mecanismo de seguridad

```
1  import inet.common.INETDefs;
2  import inet.common.packet.chunk.Chunk;
3
4  namespace inet::prueba;
5  //
6  // TODO generated message class
7  //
8
9  enum AppNodePacketType
10 {
11     JOIN_REQUEST = 1;
12     JOIN_REPLY = 2;
13     DATA = 3;
14     TXCONFIG = 4;
15 }
16
17
18 class AppNodePacket extends FieldsChunk {
19     int msgType @enum(AppNodePacketType);
20     string data;
21     string iv;            // para descifrar
22     string publicKey;     // clave pública
23     string signature;     // opcional, por si quieres verificar también
24 }
```

3.7.3.1 Mensaje base (NS): campos mínimos y propósito

El perfil base (NS) utiliza un mensaje de aplicación con campos mínimos orientados a soportar control operativo y transporte de carga útil en claro. Típicamente, el mensaje incluye un identificador de tipo (por ejemplo, para discriminar JOIN/DATA/control) y el campo de datos que porta el contenido. Este formato, deliberadamente simple, constituye la línea base para comparar escenarios y aislar el efecto de seguridad, manteniendo constante la semántica del tráfico.

3.7.3.2 Tipos de mensaje y semántica operacional (JOIN, JOIN_REPLY y DATA)

El protocolo de aplicación implementado distingue tres tipos de mensaje que estructuran el ciclo operativo del escenario: JOIN, JOIN_REPLY y DATA. JOIN representa la solicitud de incorporación del sensor hacia el gateway y habilita que el sistema establezca el estado operativo del nodo (admisión). JOIN_REPLY constituye la respuesta del gateway confirmando la incorporación y, cuando corresponde, formaliza la transición del nodo hacia la fase de transmisión regular. Finalmente, DATA representa el mensaje de carga útil periódica (sensado o información operativa) que se envía durante la fase estable de comunicación.

Bajo el perfil base (NS), estos mensajes circulan en formato mínimo, manteniendo el contenido en claro. Bajo el perfil seguro (S), los mismos tipos se conservan (para no alterar la semántica del tráfico), pero el contenido de aplicación se encapsula: el payload de JOIN y DATA se transmite protegido, y el gateway aplica validación previa para decidir admisión/entrega. Esta separación permite evaluar seguridad sin cambiar la lógica de protocolo, y a la vez instrumentar métricas asociadas a convergencia (JOIN exitosos), descartes por verificación y desempeño de entrega de DATA. La **Tabla 33** resume los tipos de mensaje implementados y su propósito operacional dentro del ciclo join → transmisión periódica.

Tabla 33

Tipos de mensaje implementados y propósito operacional.

Tipo	Emisor → Receptor	Propósito	Relevancia en seguridad
JOIN	Sensor → Gateway	Solicitud de incorporación / inicio de sesión operativa	Bajo S, se valida (verify) antes de admitir
JOIN_REPLY	Gateway → Sensor	Confirmación de incorporación / transición a fase DATA	Puede informar estado; no cambia semántica entre perfiles
DATA	Sensor → Gateway	Transporte de carga útil periódica	Bajo S, contenido protegido + verificación previa

En esta investigación, se considera que un nodo ha alcanzado convergencia operativa cuando recibe un JOIN_REPLY válido desde el gateway, confirmando su incorporación al dominio de comunicación y habilitando el inicio de la fase de transmisión de DATA bajo el perfil correspondiente (NS o S).

3.7.3.3 Mensaje seguro (S): extensión del formato y campos criptográficos

El perfil seguro (S) extiende el formato base incorporando los campos necesarios para operar el encapsulado híbrido. En lugar de transportar el payload en claro, el mensaje contiene el ciphertext producido por el cifrado simétrico y adjunta parámetros de operación (por ejemplo, IV/nonce para CTR). Adicionalmente, se incluye el componente público asociado a la clave efímera (R) para habilitar la reconstrucción del secreto compartido y/o

validaciones, y una firma ECDSA que autentica el ciphertext y metadatos mínimos. El objetivo de esta extensión es garantizar confidencialidad en tránsito y permitir aceptación condicionada mediante verificación previa, en coherencia con el modelo de amenaza.

La **Tabla 34** resume el propósito de cada campo del paquete y su contribución al overhead de transmisión cuando se habilita el perfil con seguridad.

Tabla 34

Campos del mensaje seguro y propósito.

Campo (S)	**Propósito**	**Impacto (tamaño/procesamiento)**
PublicKey (clave pública ECDH del emisor, comprimida y serializada en HEX)	Material público asociado al intercambio ECDH; habilita derivación de secreto compartido	Aumenta tamaño; habilita derivación de clave en la recepción
IV/nonce	Parámetro requerido por AES-CTR; unicidad por mensaje	Aumenta tamaño; condición crítica de confidencialidad
ciphertext	Payload cifrado (confidencialidad)	Tamaño ligado al payload; requiere cifrado/descifrado
firma ECDSA	Autenticidad/integridad; soporta verify → decrypt	Aumenta tamaño; costo de firmado/verificación; habilita descarte temprano
metadatos (id emisor, contador/timestamp si aplica)	Identidad y soporte de frescura/anti-replay según perfil	Aumenta tamaño; validaciones adicionales; mejora trazabilidad

Elementos del encapsulado ECIES a nivel de aplicación y su implicación cualitativa en tamaño y procesamiento.

3.7.3.4 Implicaciones del formato en overhead y desempeño (sin números)

El mensaje seguro introduce overhead por dos vías: incremento del tamaño transmitido y aumento del procesamiento por operación criptográfica. En un canal acústico, este overhead puede reflejarse en mayor ocupación del medio, incremento de colas y mayor sensibilidad a contención en escenarios densos. Estas implicaciones son metodológicamente relevantes porque permiten evaluar el costo incremental de seguridad de forma comparable: el canal y la pila se mantienen constantes, y la diferencia entre perfiles se concentra en el encapsulado y validación del mensaje. La **Tabla 35** detalla los campos incorporados en el paquete AppNodePacket y su rol dentro del encapsulado ECIES-based, base para interpretar el overhead sin introducir aún valores numéricos.

Tabla 35

Campos del paquete AppNodePacket y rol dentro del esquema ECIES-based

Campo	Tipo	Uso en la propuesta	Observación
msgType	int	Identifica JOIN/DATA/TXCONFIG.	Permite control de flujo por aplicación.
data	string	Payload; en perfil S transporta ciphertext (HEX).	En perfil NS transporta texto en claro.
iv	string	IV/nonce para AES-CTR (HEX).	Debe ser único por mensaje bajo una misma clave.

publicKey	string	Clave pública comprimida del emisor (HEX).	Se usa para verificación de firma y/o ECDH.
signature	string	Firma ECDSA-SHA256 del ciphertext (HEX).	Permite validación previa al descifrado.

En el entorno simulado no se implementa una infraestructura PKI ni distribución dinámica de certificados. Por ello, la confianza práctica se establece mediante el proceso de admisión (JOIN) y el registro de nodos aceptados en el gateway, complementado por una lista de bloqueo para emisores no admitidos. En este marco, el campo publicKey se interpreta como material público operativo: habilita la derivación ECDH y la verificación de firma ECDSA del emisor, pero la pertenencia a la red (legitimidad) se decide por la admisión y las reglas de control del escenario.

La estructura del mensaje de aplicación utilizado en la simulación se evidencia en la definición formal del paquete AppNodePacket, también para verlo de manera gráfica en la

Figura 37 donde se especifican los campos necesarios para operar el perfil seguro: data, iv, publicKey y signature. Esta definición constituye la referencia primaria del formato del mensaje en el modelo.

Figura 37

Parámetros de configuración de seguridad y gestión de claves en FullApp y Gwapp

Formato del mensaje de aplicación AppNodePacket

(a) AppNodePacket.msg (definición formal)

```
import inet.common.INETDefs;
import inet.common.packet.chunk.Chunk;

namespace inet::prueba;
//
// TODO generated message class
//

enum AppNodePacketType
{
    JOIN_REQUEST = 1;
    JOIN_REPLY = 2;
    DATA = 3;
    TXCONFIG = 4;
}


class AppNodePacket extends FieldsChunk {
    int msgType @enum(AppNodePacketType);
    string data;
    string iv;              // para descifrar
    string publicKey;       // clave pública
    string signature;       // opcional, por si quieres verificar también
}
```

(b) Log: payload seguro y tamaños (bytes)

```
Initializing module T3.Gateway.uanNic.radio, stage 21
INFO: Initialized (inet::Uan::UanTransducer)radio id=64, antenna =
** Event #1  t=1  T3.nodes[0].app[0] on mainLoop
** Event #2  t=1  T3.nodes[1].app[0] on mainLoop
** Event #3  t=1  T3.Gateway.app[0] on mainLoop
** Event #4  t=1.592844617  T3.nodes[0].app[0] on joinRequest
INFO: Payload 0
INFO: Data size: 8 bytes
INFO: IV size: 32 bytes
INFO: PublicKey size: 66 bytes
INFO: Signature size: 128 bytes
INFO: Tamaño total del payload 244 bytes
** Running in Express mode from event #4  t=1.592844617 ...
** Leaving Express mode at event #15616  t=1667
** Event #15617  t=1667  T3.Gateway.app[0] on mainLoop
** Event #15618  t=1667  T3.nodes[0].uanNic.queue on DataFrame
```

Nota: (a) FullApp: gwpass/nodepass, useIndexAsKey y cifrar. (b) Gwapp: gwpass y cifrar.

Por razones de compatibilidad y trazabilidad dentro de OMNeT++, los campos criptográficos del mensaje seguro se serializan como cadenas en formato HEX (iv, publicKey, signature y data). Si bien esta decisión incrementa el tamaño transmitido respecto a una codificación binaria compacta, facilita la inspección del tráfico, la depuración y el registro reproducible de los campos; además, permite cuantificar explícitamente el overhead como parte de las métricas experimentales.

3.7.5.6 Construcción del encapsulado seguro en FullApp (emisión)

La adaptación del esquema híbrido basado en ECIES se materializa en la aplicación emisora mediante una rutina de encapsulado que transforma el payload en un mensaje seguro antes de entregarlo al flujo normal de transmisión. En FullApp, el proceso se inicia con la generación del contenido (JOIN o DATA) y la decisión de perfil según la configuración (cifrar). Cuando el perfil seguro está habilitado, la aplicación ejecuta secuencialmente: (i) obtención del material de clave local (identidad del nodo) y del material público del receptor (gateway), (ii) derivación de un secreto compartido mediante acuerdo ECDH, (iii) derivación de clave simétrica con una KDF basada en hash, (iv) generación de un IV/nonce por mensaje y cifrado del payload con AES-CTR, y (v) firma ECDSA del contenido protegido y

metadatos mínimos. Finalmente, el resultado se empaqueta en el formato de aplicación, colocando el ciphertext en el campo de datos y adjuntando los campos criptográficos necesarios (IV, firma y clave pública del emisor cuando aplica).

Este bloque constituye la evidencia central de la “adaptación”: el algoritmo no se queda en un planteamiento conceptual, sino que se integra como una transformación ejecutada por la aplicación, preservando el modelo de canal y habilitando medición consistente del overhead y de eventos de aceptación en la recepción, esto puede resumirse en la **Figura 38** .

Figura 38

Evidencia de implementación del encapsulado seguro y definición del formato del mensaje de aplicación.

FullApp.cc — encapsulado (AES-CTR) y firma/verificación (ECDSA)

```
            pktRequest->setName("JsonReply");
            pktRequest->setKind(JOIN_REPLY);
            payload->setMsgType(JOIN_REPLY);
            break;

        case DATA:
        {
            pktRequest->setName("DataFrame");
            pktRequest->setKind(DATA);
            payload->setMsgType(DATA);

            if (!isCoordinator)
            {
                std::string s = "NODO: " + std::to_string(getParentModule()->getIndex()) + " DATA:" + std::to_string(counter);
                counter++;
                payload->setData(s.c_str());
            }
        }
        break;

        case TXCONFIG:
            pktRequest->setName("TxConfig");
            pktRequest->setKind(TXCONFIG);
            payload->setMsgType(TXCONFIG);
            break;

        default:
            EV_WARN << "Tipo de mensaje desconocido: " << type << endl;
            delete pktRequest;
            return;
        }

        if (cifrar)
        {
            if (!isCoordinator)
            {
                auto [ivHex, cipherHex] = aesEncrypt(privKeyBin, GwPubKeyBin, payload->getData());
                payload->setIv(ivHex.c_str());
                payload->setData(cipherHex.c_str());
                payload->setPublicKey(IntToHex(pubKeyBin, pubKeyBin.size()).c_str());
                std::string signature = signMessage(payload->getData(), privKeyBin);
                payload->setSignature(signature.c_str());
            }
            else
            {

                auto [ivHex, cipherHex] = aesEncrypt(privKeyBin, nodePubKeyBin, payload->getData());
                payload->setIv(ivHex.c_str());
                payload->setData(cipherHex.c_str());
                std::string signature = signMessage(payload->getData(), privKeyBin);
                payload->setSignature(signature.c_str());
            }
        }

        // Tags para la MAC y protocolo
        pktRequest->addTagIfAbsent<PacketProtocolTag>()->setProtocol(&Protocol::unknown);
        pktRequest->addTagIfAbsent<MacAddressReq>()->setDestAddress(macDst());

        std::string dataStr = payload->getData();
        std::string ivStr = payload->getIv();
        std::string pubKeyStr = payload->getPublicKey();
        std::string sigStr = payload->getSignature();

```

AppNodePacket.msg — definición del formato (campos criptográficos)

```
import inet.common.INETDefs;
import inet.common.packet.chunk.Chunk;

namespace inet::prueba;
//
// TODO generated message class
//

enum AppNodePacketType
{
    JOIN_REQUEST = 1;
    JOIN_REPLY = 2;
    DATA = 3;
    TXCONFIG = 4;
}


class AppNodePacket extends FieldsChunk {
    int msgType @enum(AppNodePacketType);
    string data;
    string iv;          // para descifrar
    string publicKey;   // clave publica
    string signature;   // opcional, por si quieres verificar tambien
}
```

Nota**.** (a) Fragmento de *FullApp.cc* donde se realiza el cifrado AES-CTR, la firma/verificación ECDSA y la asignación de campos (*data*, *iv*, *publicKey*, *signature*) en el paquete. (b) Definición de *AppNodePacket.msg* con los campos utilizados para transportar el ciphertext y metadatos criptográficos.

3.7.5.7 Reconstrucción y validación en Gwapp (recepción)

En el gateway, el procesamiento del mensaje seguro se implementa con una lógica complementaria orientada a validación temprana. Al recibir un paquete seguro, Gwapp recupera la clave pública del emisor desde el propio mensaje y ejecuta la verificación ECDSA sobre el contenido protegido. Solo si la verificación es satisfactoria, el gateway deriva el secreto compartido mediante ECDH utilizando su clave privada local y la clave pública recibida, aplica la misma KDF para reconstruir la clave simétrica y descifra el ciphertext con AES-CTR empleando el IV recibido. El payload resultante se entrega a la lógica de aplicación (JOIN para admisión o DATA para registro), mientras que los mensajes inválidos se descartan registrando el evento como fallo de validación, coherente con la política verify → decrypt. En la **Figura 39** el gateway implementa una política fail-fast para el ingreso (JOIN) en modo seguro: primero reconstruye la identidad pública del emisor a partir del campo publicKey y valida la firma ECDSA asociada al mensaje recibido. Únicamente si la verificación es satisfactoria se procede a derivar el material de clave mediante ECDH+KDF (implícito en la rutina aesDecrypt) y a descifrar el contenido con AES-CTR. Este orden operacional permite descartar mensajes no auténticos sin incurrir en el costo de descifrado, y mantiene coherencia con el enfoque “verificar → descifrar” definido para la evaluación experimental.

Figura 39

Procesamiento del JOIN en el gateway (Gwapp) bajo perfil seguro: recuperación de la clave

pública del emisor (publicKey), verificación de firma ECDSA previa al descifrado y posterior derivación de clave

```
if (nodosUnidos.find(srcMac) == nodosUnidos.end())
{
    if (cifrar)
    {
        auto nodePubKeyBin = HexToSecByteBlock(payload->getPublicKey());
        auto a = verifySignature(payload->getData(), payload->getSignature(), payload->getPublicKey());

        if (a)
        {
            auto b = payload->getData();
            auto c = aesDecrypt(privKeyBin, nodePubKeyBin, payload->getIv(), b);
            if (c == "JOIN")
            {
                nodosUnidos[srcMac] = 1;
                EV_INFO << "Nodo registrado: " << srcMac << endl;

                if (std::find(fifoQueueJoin.begin(), fifoQueueJoin.end(), srcMac) == fifoQueueJoin.end())
                {
                    fifoQueueJoin.push_back(srcMac); // Solo lo agrega si no existe
                }
            }
            else
            {

                EV_INFO << "Nodo no autentificado: " << srcMac << endl;
```

3.7.4 Parámetros principales de configuración (INI)

Los perfiles se ejecutan mediante secciones del archivo INI, que controlan parámetros como "numberOfNodes", "sim-time-limit" y la activación del bloque criptográfico mediante "cifrar". En el perfil S, además de cifrar el payload, se genera la firma ECDSA y se adjuntan los campos iv/publicKey/signature. La **Tabla 24** (Sección 3.5) consolida la equivalencia entre perfiles y secciones INI utilizadas en la campaña de simulación.

3.7.4.1 Tasa de transmisión configurada (1200 bps)

En la configuración del escenario se establece una tasa de transmisión de 1200 bps (≈1.2 kbps) como valor representativo de enlaces acústicos de baja tasa en UWSN. La literatura clásica de redes acústicas subacuáticas destaca que la disponibilidad de ancho de banda es severamente limitada y que, como consecuencia, los dispositivos existentes operan con bit rates bajos, condicionados por rango y frecuencia; en particular, se reportan ejemplos de enlaces en el orden de kilobits por segundo, incluyendo valores alrededor de 1.2 kbps en tablas comparativas de técnicas/modems utilizados en comunicaciones acústicas. Además, en un trabajo de referencia sobre plataformas acústicas ampliamente utilizadas en investigación

(Akyildiz et al., 2005b; Gallimore et al., 2010) se documenta que los modos operativos contemplan tasas desde decenas de bps hasta algunos kbps, indicando rangos de operación reportados de 80 a 5000 bps.

Con base en lo anterior, 1200 bps se adopta como un punto de operación conservador y realista para escenarios submarinos donde la capacidad del canal es restringida. Esta elección fortalece el control experimental porque: (i) mantiene constante la capacidad física entre perfiles (NS/S), y (ii) hace observable el impacto del encapsulado seguro (sobrecarga en tamaño y tiempo de transmisión), sin asumir condiciones de alta tasa que no son típicas en comunicaciones acústicas subacuáticas.

3.7.5 Integración del bloque criptográfico en el flujo de aplicación

El punto de integración del mecanismo se ubica en la lógica de aplicación de los nodos (FullApp) y del gateway (Gwapp). Cuando el parámetro "cifrar" está habilitado, el emisor: (i) deriva el secreto ECDH, (ii) calcula SHA-256 del secreto, (iii) toma 16 bytes para AES-128, (iv) genera un IV aleatorio de 16 bytes, (v) cifra el payload con AES-CTR y (vi) firma el ciphertext mediante ECDSA-SHA256. En la recepción, primero se verifica la firma usando la clave pública recibida; solo si la firma es válida se procede al descifrado. Este comportamiento asegura consistencia con el marco teórico y evita procesamiento de tráfico inválido antes de consumir recursos en el descifrado (NIST, 2023a; NIST, 2023b). La firma digital se calcula sobre una representación canónica del mensaje protegido, incluyendo el tipo de mensaje y los parámetros críticos de operación. La autenticidad e integridad se garantizan mediante firma ECDSA calculada sobre el contenido cifrado (ciphertext), de modo que el receptor pueda validar el paquete antes de considerar su procesamiento. En el marco implementado, la firma se aplica al campo data (que transporta el ciphertext codificado), mientras que el IV se emplea como parámetro de descifrado en AES-CTR y la clave pública

del emisor se utiliza como material público operativo para ECDH y verificación de firma en el receptor.

3.7.5.1 Construcción del bloque ECIES dentro del código del proyecto

La integración del mecanismo de seguridad se materializa como un bloque criptográfico embebido en las aplicaciones principales del escenario (FullApp y Gwapp), implementado mediante funciones dedicadas a: (i) derivación de claves por nodo a partir de parámetros de configuración, (ii) acuerdo ECDH para obtención de secreto compartido, (iii) derivación de clave simétrica (KDF basada en SHA-256), (iv) cifrado/descifrado con AES-CTR y (v) firma/verificación ECDSA para control de integridad y autenticidad. Esta organización permite encapsular la seguridad como una transformación del payload sin alterar la pila base, haciendo que el efecto de seguridad sea observable y medible desde la instrumentación del escenario.

3.7.5.2 Emisor: encapsulado seguro y envío

Cuando el perfil seguro está habilitado mediante el parámetro cifrar, el emisor realiza el encapsulado de forma consistente para cada paquete de aplicación. El proceso inicia a partir del payload en claro generado por la aplicación y procede con un acuerdo ECDH entre la clave privada local y la clave pública del par (gateway o nodo destino, según el rol). El secreto compartido obtenido se transforma mediante SHA-256 para derivar material simétrico, del cual se extrae la clave usada por AES-CTR. Para garantizar unicidad por mensaje y evitar reutilización del flujo, el emisor genera un IV aleatorio por paquete y cifra el payload, produciendo el ciphertext.

Una vez cifrado el contenido, el emisor genera una firma ECDSA sobre el ciphertext (representado en el campo data) y adjunta la firma al mensaje. En el flujo nodo→gateway, además se adjunta la clave pública del emisor en formato comprimido para facilitar verificación y reconstrucción de secreto compartido en el receptor. Finalmente, el paquete se

transmite por el mismo canal y pila base del escenario, preservando la comparabilidad experimental entre perfiles. Como se evidencia en el código de FullApp, cuando el perfil seguro (cifrar) está habilitado, el emisor encapsula cada mensaje adjuntando el IV, el ciphertext y la firma ECDSA (y, en el flujo nodo→gateway, también la clave pública del emisor), tal como se resume en la **Figura 40**.

Figura 40

Encapsulado seguro en el emisor (FullApp)

```
if (cifrar)
{
    if (!isCoordinator)
    {
        auto [ivHex, cipherHex] = aesEncrypt(privKeyBin, GwPubKeyBin, payload->getData());
        payload->setIv(ivHex.c_str());
        payload->setData(cipherHex.c_str());
        payload->setPublicKey(IntToHex(pubKeyBin, pubKeyBin.size()).c_str());
        std::string signature = signMessage(payload->getData(), privKeyBin);
        payload->setSignature(signature.c_str());
    }
    else
    {

        auto [ivHex, cipherHex] = aesEncrypt(privKeyBin, nodePubKeyBin, payload->getData());
        payload->setIv(ivHex.c_str());
        payload->setData(cipherHex.c_str());
        std::string signature = signMessage(payload->getData(), privKeyBin);
        payload->setSignature(signature.c_str());
    }
}
```

Nota: con cifrar habilitado se genera (IV, ciphertext) mediante aesEncrypt(...) y se adjuntan iv y data. En el flujo nodo→gateway se incluye además publicKey y la firma ECDSA (signature) sobre el ciphertext.

3.7.5.3 Receptor: validación, descarte y entrega

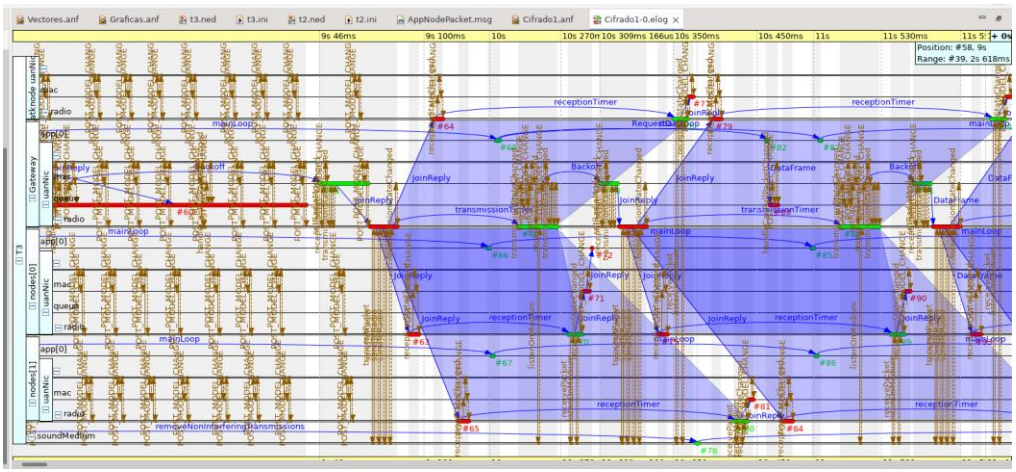
En la implementación, la política operativa se aplica de forma diferenciada según el tipo de mensaje y el rol del nodo. En la fase de admisión (JOIN), el gateway ejecuta una validación estricta: únicamente si la firma es válida se procede al descifrado y a la aceptación del nodo. En el intercambio de datos (DATA), la verificación se mantiene como control de autenticidad y como señal instrumentada para el análisis; en particular, en los nodos sensores la entrega/activación de respuesta se condiciona a la verificación, mientras que en el gateway

la verificación se registra y se complementa con controles de admisión previamente establecidos. Esta decisión se adopta para conservar el control experimental del escenario y separar claramente eventos de admisión, tráfico y medición, quedando el descarte estricto por firma inválida en DATA como extensión directa del mismo principio.

Una vez superada la validación (especialmente crítica en el proceso de admisión/join), el receptor reconstruye el secreto compartido mediante ECDH, deriva la clave simétrica con SHA-256 y ejecuta el descifrado AES-CTR utilizando el IV recibido. El payload recuperado se entrega a la lógica de aplicación como dato válido, manteniendo el resto del flujo (cola, temporización y transmisión) sin cambios respecto al perfil base, esto se observa en la **Figura 41**.

Figura 41

Diagrama de secuencia conceptual del envío seguro y la recepción con política verify → decrypt (ECIES: ECDH/KDF + AES-CTR + firma ECDSA).



Para evidenciar que la política operativa del gateway se implementa conforme al flujo verify → decrypt, se presenta un fragmento representativo del módulo Gwapp en el que se procesa la recepción de mensajes desde la capa inferior. En modo seguro, el gateway extrae del paquete los campos criptográficos necesarios (p. ej., publicKey, signature e iv), ejecuta

primero la validación de firma ECDSA sobre el ciphertext y metadatos, y únicamente si la verificación es exitosa procede al descifrado AES-CTR derivando el secreto mediante ECDH. Este orden permite descartar tráfico manipulado antes de consumir cómputo en descifrado o alterar el estado de admisión del nodo. La **Figura 42** ilustra explícitamente esta secuencia de aceptación/rechazo en el caso de mensajes de tipo JOIN.

Figura 42

Recepción segura en el gateway y política de aceptación verify → decrypt (ECDSA previa y descifrado AES-CTR con secreto ECDH derivado).

```
if (cifrar)
{

    auto nodePubKeyBin = HexToSecByteBlock(payload->getPublicKey());
    auto a = verifySignature(payload->getData(), payload->getSignature(), payload->getPublicKey());

    if (a)
    {
        auto b = payload->getData();
        auto c = aesDecrypt(privKeyBin, nodePubKeyBin, payload->getIv(), b);
        if (c == "JOIN")
        {
            nodosUnidos[srcMac] = 1;
            EV_INFO << "Nodo registrado: " << srcMac << endl;

            if (std::find(fifoQueueJoin.begin(), fifoQueueJoin.end(), srcMac) == fifoQueueJoin.end())
            {
                fifoQueueJoin.push_back(srcMac); // Solo lo agrega si no existe
            }
        }
        else
        {

            EV_INFO << "Nodo no autentificado: " << srcMac << endl;

            return;
```

Nota. Fragmento de gwapp.cc en Gwapp::handleMessageFromLowerLayer(), caso JOIN_REQUEST (modo cifrado).

Por razones de compatibilidad y trazabilidad en OMNeT++, los campos criptográficos del mensaje seguro se serializan como cadenas en formato HEX (iv, publicKey, signature y el data cifrado). Esta decisión facilita inspección de tráfico, depuración y registro, pero incrementa el tamaño transmitido respecto a formatos binarios compactos.

Metodológicamente, este incremento es relevante porque se convierte en una fuente controlada de overhead que el diseño experimental puede cuantificar y contrastar entre perfiles. Como soporte del formato adoptado para el intercambio seguro, la **Figura 43** presenta la definición del mensaje AppNodePacket, donde se observa que los campos criptográficos (data, iv, publicKey y signature) se declaran explícitamente y se serializan como cadenas en formato HEX para facilitar trazabilidad y depuración durante la simulación.

Figura 43

Definición del mensaje seguro AppNodePacket: campos data, iv, publicKey y signature (serialización HEX).

```
16
17
18 class AppNodePacket extends FieldsChunk {
19     int msgType @enum(AppNodePacketType);
20     string data;
21     string iv;            // para descifrar
22     string publicKey;     // clave pública
23     string signature;     // opcional, para verificar
24 }
```

3.7.5.4 Gestión de claves durante ejecución simulada

Durante la ejecución del perfil seguro, cada entidad legítima opera con material de clave consistente con su rol, conforme a la gestión de claves definida en la arquitectura. La identidad criptográfica se expresa mediante la asociación entre identificadores de nodo y claves públicas utilizadas para validación, mientras que la clave efímera generada por el emisor habilita la obtención de secretos de sesión asociados a cada transmisión segura. Bajo este enfoque, la simulación mantiene el supuesto de disponibilidad de las claves públicas requeridas, evitando introducir variabilidad no controlada en la distribución de claves y concentrando la evaluación en el costo e impacto del encapsulado.

La gestión de claves se operacionaliza mediante configuración del escenario: el gateway y los nodos derivan su material criptográfico desde parámetros (gwpass, nodepass) y pueden diferenciar identidad por nodo mediante useIndexAsKey. Con ello, cada entidad dispone de una clave privada local y una clave pública asociada que se utiliza tanto para el acuerdo ECDH como para verificación ECDSA. En el flujo nodo→gateway, la clave pública del emisor se incorpora al mensaje para permitir verificación y derivación del secreto sin requerir mecanismos externos de distribución dinámica, manteniendo el control experimental sobre el entorno.

Este enfoque no persigue modelar una infraestructura completa de certificación, sino representar un dominio experimental donde las identidades criptográficas son consistentes con la configuración del escenario y con las políticas de admisión/observación definidas para los perfiles NS y S.

3.7.5.5 Instrumentación habilitada para Cap. 4

La integración del bloque criptográfico se acompaña de instrumentación orientada a medir, de forma comparable entre perfiles, el costo y el efecto de seguridad sobre la comunicación. Se registran: (i) tiempos de operaciones criptográficas (cifrado/descifrado y firmado/verificación), (ii) eventos de aceptación y descarte por validación, (iii) variación del tamaño lógico del mensaje entre NS y S, y (iv) métricas de red sensibles al overhead (latencia extremo a extremo, PDR/PLR y tasa efectiva de entrega). Con ello, el Capítulo 4 puede reportar resultados agregados por densidad y perfil, cuantificando el impacto relativo de incorporar ECIES en la red acústica simulada y evidenciando el comportamiento ante presencia de atacantes cuando el escenario lo contempla.

Con la arquitectura del escenario definida, el medio acústico fijado como componente invariante (UanSoundMedium) y el bloque de seguridad integrado a nivel de aplicación con política verify → decrypt, el sistema queda especificado desde el punto de vista de diseño e

implementación. No obstante, antes de presentar resultados cuantitativos, es necesario describir la fase operativa que garantiza que los perfiles (NS, S y escenarios con atacante) se ejecutan de manera consistente, y que la instrumentación produce salidas comparables. Por ello, la siguiente sección documenta las pruebas de verificación funcional y la preparación de campañas que sirven como base directa para el Capítulo 4.

3.8 Pruebas de verificación y preparación de campañas de evaluación

3.8.1 Propósito y alcance de la fase de pruebas

Esta sección consolida la fase operativa previa a la evaluación cuantitativa: se describen las pruebas empleadas para confirmar que (i) la comunicación base en OMNeT++ opera de forma estable, (ii) el encapsulado de seguridad se integra como transformación del mensaje en la capa de aplicación, y (iii) el comportamiento bajo atacante se ajusta al modelo de amenazas definido. En consecuencia, el objetivo de estas pruebas no es reportar desempeño (lo cual corresponde al Capítulo 4), sino asegurar que el sistema está correctamente instrumentado y que los escenarios pueden ejecutarse bajo condiciones repetibles y comparables.

De acuerdo con la validación incremental utilizada durante la implementación, la verificación se organiza en tres niveles: primero, un escenario base sin seguridad (NS) para validar conectividad y formato; segundo, un escenario con seguridad (S) para confirmar la política verify → decrypt; y tercero, escenarios con atacante para observar descartes, intentos de perturbación y consistencia del control de admisión sin asumir ruptura criptográfica.

3.8.2 Configuración de escenarios experimentales (6 condiciones)

La preparación de campañas se estructura en seis escenarios definidos por configuración, preservando el canal acústico y la pila base como elementos invariantes (Sección 3.4). Cada escenario combina seguridad (NS/S) y tipo de atacante (Normal/Atacante/Listener):

- • NS_Normal: sin seguridad, sin atacante.

  • NS_Atacante: sin seguridad, con atacante activo.

  • NS_Listener: sin seguridad, con listener pasivo.

  • S_Normal: con seguridad, sin atacante.

  • S_Atacante: con seguridad, con atacante activo.

  • S_Listener: con seguridad, con listener pasivo.

Esta organización permite que el Capítulo 4 compare escenarios equivalentes donde cambian únicamente los factores experimentales (seguridad y/o atacante), evitando sesgos por modificaciones del canal o de la pila base, también lo detallamos en la **Tabla 39**.

Tabla 38

Escenarios experimentales y factores que varían por configuración (6 condiciones).

Escenario	**Seguridad (ECIES)**	**Política de aceptación**	**Atacante**	**Propósito metodológico**
NS_Normal	Deshabilitada	N/A (mensaje base)	No	Línea base de comunicación y tráfico.
NS_Atacante	Deshabilitada	N/A (mensaje base)	Atackapp (activo)	Medir degradación por ataque en ausencia de seguridad.
NS_Listener	Deshabilitada	N/A (mensaje base)	Listener (pasivo)	Discutir exposición del

				payload bajo escucha.
S_Normal	Habilitada	verify → decrypt	No	Medir costo y efectos del encapsulado seguro.
S_Atacante	Habilitada	verify → decrypt	Atackapp (activo)	Evaluar robustez operativa y trade-off bajo ataque.
S_Listener	Habilitada	verify → decrypt	Listener (pasivo)	Discutir confidencialidad: escucha no debe recuperar payload.

3.8.3 Verificación funcional del ciclo JOIN → JOIN_REPLY → DATA

La verificación funcional del sistema se apoya en el protocolo de aplicación descrito en 3.7.3: el comportamiento esperado del escenario se organiza en el ciclo JOIN → JOIN_REPLY → DATA, donde la convergencia operativa se define cuando el nodo recibe un JOIN_REPLY válido desde el gateway. Bajo esta definición, la fase de pruebas confirma que:

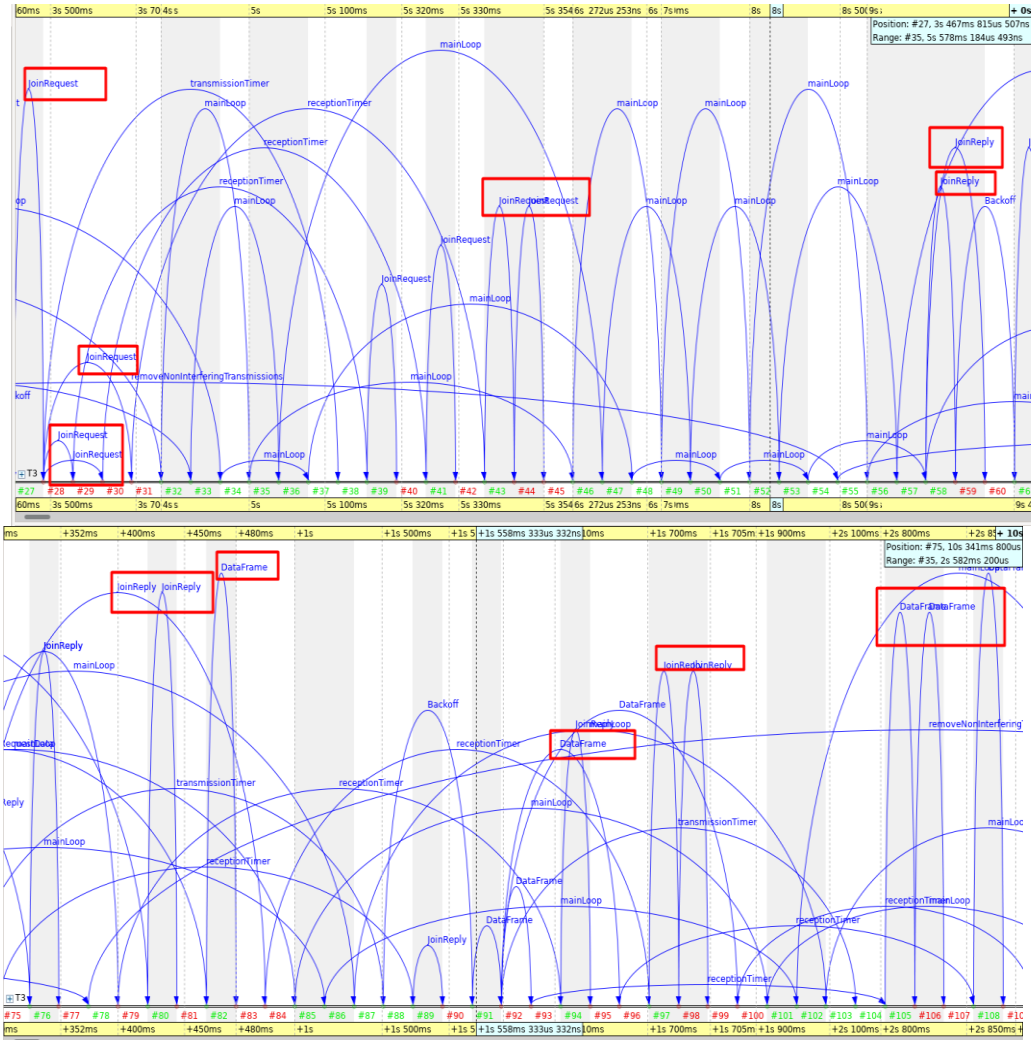
1. El sensor inicia incorporación mediante JOIN.

2. El gateway responde con JOIN_REPLY y registra la admisión.
3. El nodo, tras converger, inicia el envío periódico de DATA y el gateway procesa recepción/entrega de acuerdo con el perfil activo.

En la **Figura 44** en el perfil NS, la verificación se centra en conectividad, temporización y consistencia del formato base. En el perfil S, se comprueba que el ciclo se conserva semánticamente, pero el contenido viaja encapsulado, y que la aceptación queda condicionada por validación criptográfica.

Figura 44

Traza temporal del ciclo operativo de la aplicación: JOIN → JOIN_REPLY → DATA en el escenario base.



3.8.4 Verificación del flujo seguro: encapsulado y política verify → decrypt

Para el perfil seguro, la verificación confirma que el mecanismo descrito en 3.7.5 opera como transformación del mensaje a nivel de aplicación y que el receptor ejecuta la política verify → decrypt de forma determinista. En esta etapa se valida que el emisor construye el paquete seguro con los campos definidos (por ejemplo iv, data como ciphertext, signature y publicKey cuando corresponda), y que el receptor descarta mensajes inválidos sin ejecutar descifrado, registrando el evento como fallo de validación.

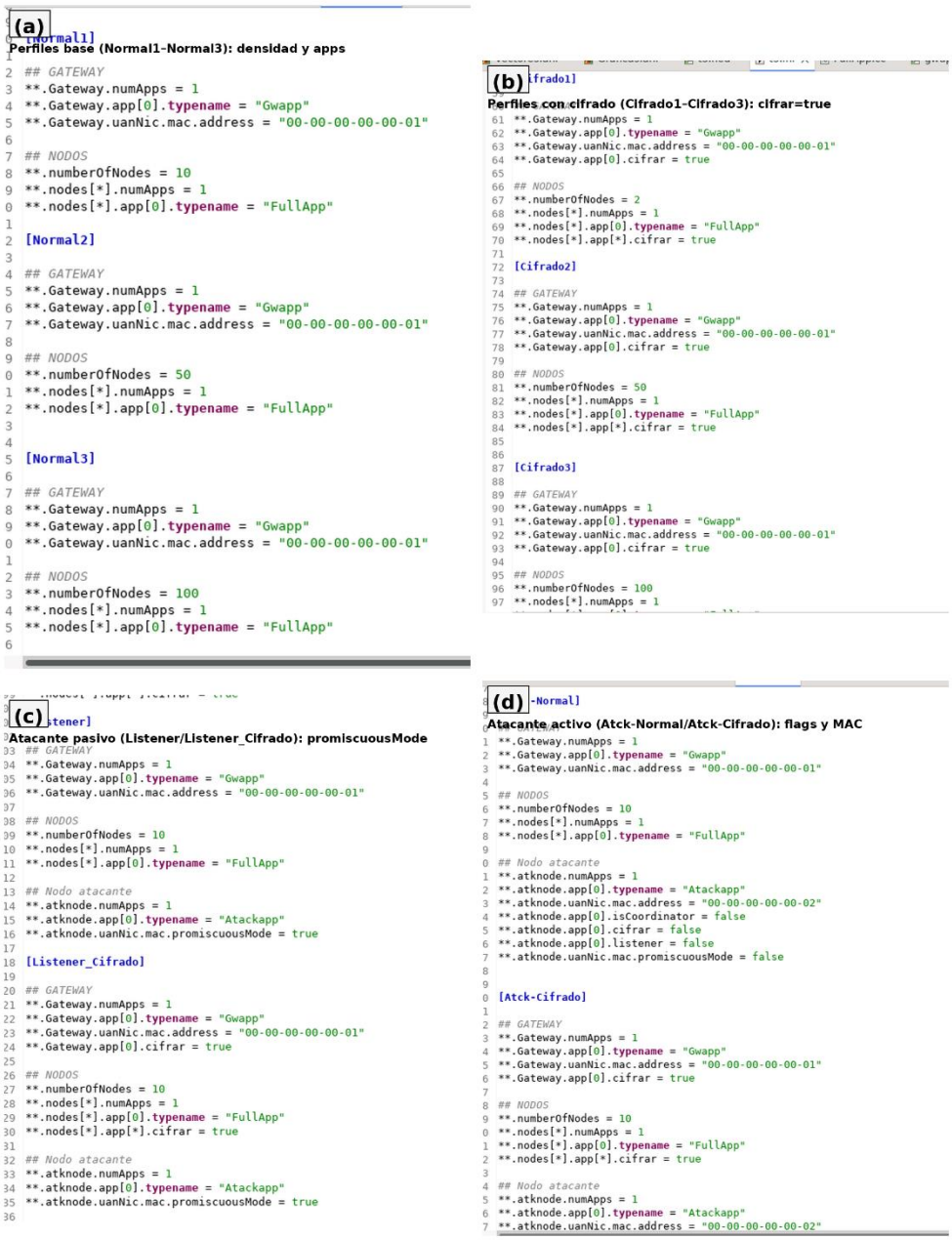
Esta comprobación es relevante porque separa dos fuentes distintas de “no entrega”: (i) pérdidas atribuibles al canal acústico y (ii) descartes atribuibles a seguridad por verificación fallida. Al dejar esta distinción instrumentada desde Capítulo 3, el Capítulo 4

puede interpretar correctamente PDR/PLR/goodput sin confundir degradación del canal con rechazo criptográfico.

La activación de los perfiles experimentales se realiza mediante parametrización en el archivo de configuración t3.ini. En particular, el perfil base (NS) y el perfil seguro (S) se distinguen por la bandera cifrar, mientras que los escenarios con atacante se habilitan mediante la configuración del nodo atacante (pasivo en modo promiscuo o activo con envío). La **Figura 45** presenta extractos representativos de dichas secciones, usados como ancla documental para la reproducibilidad de los experimentos.

Figura 45

Selección de perfiles experimentales en el archivo t3.ini (base, cifrado y escenarios con atacante).



Nota: (a) Perfiles base Normal1–Normal3; (b) perfiles seguros Cifrado1–Cifrado3; (c) atacante pasivo Listener/Listener_Cifrado; (d) atacante activo Atck-Normal/Atck-Cifrado.

3.8.5 Activación del atacante y verificación del comportamiento bajo amenaza

Una vez verificado el flujo base y el flujo seguro, se habilita el atacante conforme al modelo de amenazas (3.3.4). El objetivo de esta fase es confirmar que el atacante cumple su rol experimental sin introducir supuestos de ruptura criptográfica: el Listener observa tráfico

y patrones, mientras que Atackapp intenta perturbar mediante inyección/alteración, provocando fallos de verificación y descartes tempranos en el receptor.

Bajo el perfil seguro, el comportamiento esperado es que el atacante pueda capturar parámetros públicos (incluyendo claves públicas transmitidas y nonces), pero no pueda derivar la clave de sesión ni producir firmas válidas sin material privado legítimo. Por tanto, la evidencia operativa se expresa como incremento de validaciones fallidas/descartes bajo ataque, sin comprometer la confidencialidad del payload.

3.8.6 Preparación de campañas: repetibilidad, semillas e instrumentos de salida

Para asegurar que los resultados del Capítulo 4 son reproducibles y comparables, la campaña separa explícitamente dos controles: (i) la aleatoriedad propia del simulador (temporización, contención, eventos estocásticos) gobernada por seed-set, y (ii) la identidad criptográfica determinística gobernada por parámetros de configuración (gwpass, nodepass, useIndexAsKey). Ambos controles son independientes: el primero permite repetibilidad del comportamiento estocástico, y el segundo mantiene consistencia de identidad sin añadir tráfico extra de distribución de claves sobre el canal acústico.

En esta preparación, se verifica que la instrumentación registra consistentemente métricas de comunicación (entrega, retardo, tasa efectiva) junto con eventos de seguridad (verificación exitosa/fallida, descartes por validación, cambios de estado de admisión) verificado en la **Tabla 36**. Con ello, el Capítulo 4 puede basarse en salidas trazables del simulador (scalars/vectors/eventlog) bajo un marco de ejecución controlado.

Tabla 36

Diferencia entre semilla de simulación (seed-set) y material de inicialización criptográfica.

Concepto	Qué controla	Dónde se define (tu proyecto)	Qué afecta en el experimento	Justificación metodológica

Semilla de simulación (seed-set)	Aleatoriedad del simulador (temporización interna, contención/back off y otros procesos estocásticos cuando aplican)	OMNeT++ INI (opcional): parámetro global seed-set = N en [General] o configuración equivalente.	Puede introducir variación entre corridas si se repite el experimento bajo diferentes semillas.	Se usa cuando se requiere repetibilidad estadística (múltiples corridas). En tu planteamiento, la campaña se basa en configuraciones controladas y comparables, sin enfocarse en repetición masiva.
Inicialización criptográfica determinística (identidad)	Identidad criptográfica por rol (gateway, nodo, atacante) y forma de derivación	Parámetros de las Apps (NED) con opción de override en INI: • Gateway (Gwapp): **.Gateway.app[0].gwpass (default en Gwapp.ned: "admin").• Nodos (FullApp):	Define claves/identidad estables y el perfil NS/S/A sin depender de aleatoriedad del simulador.	Evita agregar protocolos de distribución dinámica de claves (tráfico extra) y preserva control experimental: las diferencias se atribuyen al

	**.nodes[*].app[0].gwpass, **.nodes[*].app[0].nodepass, **.nodes[*].app[0].useIndexAsKey (defaults en FullApp.ned: "admin", "nodo", false).• Atacante (Atackapp): **.atknode.app[0].gwpass, **.atknode.app[0].nodepass, **.atknode.app[0].useIndexAsKey (defaults en Atackapp.ned: "123", "nodo", false).• Activación de seguridad (perfil) en t3.ini: **.Gateway.app[0].cifrar,	encapsulado/validación, no a variaciones de identidad.

	**.nodes[*].app[*].ci frar, **.atknode.app[0].cif rar.

En este trabajo se prioriza la comparabilidad entre escenarios (6 condiciones) bajo un escenario base invariante. En consecuencia, se fija la configuración del simulador de manera consistente para garantizar que una ejecución represente fielmente el comportamiento del escenario definido. Cuando el entorno modela procesos pseudoaleatorios internos (por ejemplo, temporización de acceso al medio o variaciones propias del enlace), estos se mantienen controlados mediante una configuración estable, de modo que el análisis del Capítulo 4 se centre en diferencias atribuibles a seguridad y/o atacante, y no a variaciones no controladas.

3.8.7 Criterios de aceptación de pruebas y trazabilidad de evidencias

Como cierre de esta fase, se establecen criterios mínimos para considerar que un escenario está listo para evaluación cuantitativa: (i) convergencia operativa confirmada por JOIN_REPLY válido, (ii) tráfico DATA observable en fase estable, (iii) en perfil S, evidencia de verify → decrypt con descartes coherentes ante alteración, y (iv) en escenarios con atacante, incremento de eventos de fallo de verificación o perturbación sin aceptación de payload manipulado. Esta lista permite sostener que los resultados del Capítulo 4 se construyen sobre escenarios funcionalmente verificados, la **Tabla 37** detalla la lista de resultados a obtener..

Tabla 37

Criterios de verificación funcional y evidencia asociada

Criterio	Qué se valida	Evidencia principal	Salida/artefacto típico
Convergencia	JOIN → JOIN_REPLY válido	Evento de respuesta y cambio de estado	eventlog / Sequence Chart / log
Tráfico estable	Inicio y periodicidad de DATA	Mensajes DATA recibidos en gateway	scalars/vectors (conteos)
Verify → decrypt	Validación previa y descarte temprano	Mensajes rechazados sin descifrado	contadores de verificación / logs
Robustez ante atacante	Listener observa / Atackapp perturba sin romper firma	Incremento de fallos de validación	drops por verificación; trazas
Comparabilidad	Mismo canal y pila base	Configuración invariante de medio	NED del escenario; INI base

3.9 Síntesis del capítulo

En este capítulo se estableció el marco metodológico y técnico necesario para ejecutar una evaluación reproducible de un esquema de seguridad basado en ECIES sobre una red de sensores submarinos simulada en OMNeT++ 6.1. En primer lugar, se definió la arquitectura del sistema propuesto y la integración conceptual del mecanismo de seguridad (Sección 3.3),

delimitando su alcance a nivel de flujo extremo a extremo y preservando la separación entre diseño y detalles de implementación.

Posteriormente, se describió el entorno experimental (Sección 3.4), precisando la organización del modelo base y la capa de evaluación, los supuestos del canal acústico y el esquema de instrumentación que soporta la recolección de resultados. A partir de este entorno, se estructuraron los escenarios de simulación (Sección 3.5) distinguiendo escenarios de construcción/verificación (S0–S3) y una campaña de evaluación por densidad (N = 20, 50 y 100) con seis escenarios (NS/S en modos Normal/Atacante/Listener) para aislar el efecto de la seguridad y la presencia de atacante.

Finalmente, se formalizaron las métricas y el esquema de recolección y procesamiento (Sección 3.6), definiendo de manera operacional indicadores como retardo extremo a extremo, throughput/goodput, tasa de entrega de paquetes y overhead criptográfico. Con estos elementos, se garantiza que el Capítulo 4 pueda enfocarse exclusivamente en la ejecución experimental y en la presentación de resultados comparativos, sustentando conclusiones sobre el impacto de incorporar ECIES (AES-CTR + ECDSA) en UWSN bajo incrementos progresivos de densidad y condiciones de amenaza definidas.

La construcción de los incrementos S0–S3 se organizó mediante Kanban, asegurando entrega incremental, trazabilidad con requisitos y criterios de finalización antes de ejecutar la campaña de evaluación.

3.9.1 Limitaciones del modelo y alcance

La evaluación presentada en este trabajo se basa en un entorno de simulación controlado; por tanto, cualquier fenómeno que no esté explícitamente representado por los modelos y parámetros del escenario se considera una limitación del alcance. En particular, la caracterización del canal acústico se restringe a lo que el medio simulado (UanSoundMedium y la interfaz UAN asociada) modela de forma directa; efectos físicos y oceanográficos

adicionales por ejemplo, variaciones finas de salinidad y temperatura, microturbulencia, corrientes, dispersión espacial no homogénea y multipath complejo dependiente del fondo no se incorporan si no están parametrizados en el escenario.

Asimismo, aunque la simulación contempla configuraciones de distancia y profundidad como parte del diseño del escenario, la profundidad se interpreta como una variable geométrica/controlada y no como un modelo completo de estratificación oceánica (p. ej., capas con perfiles dinámicos que alteren velocidad del sonido y atenuación en función del tiempo). Por ello, los resultados deben entenderse como válidos para los rangos y condiciones configuradas, sin extrapolar automáticamente a entornos con variación ambiental intensa o perfiles de profundidad altamente dinámicos.

En el plano de seguridad, se asume un atacante con capacidades de canal (escucha e inyección/alteración) sin ruptura de primitivas criptográficas, y se utiliza inicialización determinística de identidades por configuración en lugar de una infraestructura completa de certificación, revocación y distribución dinámica de credenciales. Finalmente, cualquier componente de consumo energético o temporización que no esté instrumentado de forma explícita en el modelo se interpreta como aproximación, por lo que el análisis energético refleja el comportamiento dentro del marco de simulación y no una medición física directa.

Adicionalmente, al tratarse de un simulador de eventos discretos a nivel de paquetes, OMNeT++ no genera señales acústicas continuas; por tanto, ciertas métricas de capa física y de implementación no son observables de forma directa en este marco experimental:

Métricas de forma de onda/capa física: SNR instantáneo, Eb/N0, BER/FER, espectro/PSD, parámetros de modulación y codificación, y métricas derivadas de señal (p. ej., respuesta impulsional o coherencia temporal).

Efectos acústicos avanzados no parametrizados: Doppler dependiente de movimiento, multipath complejo dependiente del fondo/superficie, y variaciones ambientales dinámicas (salinidad/temperatura) que alteren la velocidad del sonido en el tiempo.

Métricas de ejecución en hardware real: tiempo de CPU (wall-clock), uso de memoria y consumo energético real del microcontrolador durante ECDH/AES/ECDSA; en esta tesis estos costos se representan como variables/retardos/consumos modelados en el simulador.

Capítulo 4 Resultados

El presente capítulo expone los resultados obtenidos en la campaña experimental final y su comparación sistemática entre los seis escenarios definidos: NS_Normal, NS_Atacante, S_Normal, S_Atacante, NS_Listener y S_Listener. La evaluación se realizó mediante un barrido de densidad de nodos (D20, D50 y D100; N = 20, 50 y 100), manteniendo constantes el modelo de canal acústico, la topología base, el patrón de tráfico y los parámetros físicos definidos en el Capítulo 3.

El propósito principal de esta etapa experimental es analizar cómo escala el desempeño del sistema ante el incremento de densidad y cómo se manifiesta el compromiso entre seguridad, latencia y eficiencia operativa bajo condiciones controladas. Asimismo, se busca evaluar la robustez del esquema criptográfico frente a la presencia de atacantes activos y pasivos, sin introducir modificaciones adicionales en el modelo físico del canal. De esta manera, cualquier variación observada en las métricas se atribuye exclusivamente a los factores experimentales manipulados: activación del mecanismo de seguridad, presencia o ausencia de atacante y densidad de nodos.

4.1 Configuración experimental final

Esta sección consolida la configuración experimental final utilizada para ejecutar la campaña de simulación y obtener los resultados comparativos del capítulo. La configuración se diseñó para (i) asegurar comparabilidad entre condiciones (mismo canal, topología, instrumentación y horizonte de simulación) y (ii) mantener rigor en el análisis mediante un diseño factorial con repeticiones. La densidad se evalúa en tres niveles: D20, D50 y D100 (N = 20, 50 y 100 nodos sensores), y en cada densidad se conmutan los seis escenarios (NS/S × Normal/Atacante/Listener).

La configuración experimental final se diseñó bajo dos criterios fundamentales. En primer lugar, garantizar comparabilidad entre condiciones, utilizando la misma red y el mismo

modelo de canal para todos los escenarios. En segundo lugar, asegurar rigor estadístico sin exceder el número de corridas necesarias, mediante un barrido controlado de densidad y un número uniforme de repeticiones por condición.

El mecanismo evaluado corresponde a un esquema híbrido basado en ECDH + KDF (SHA-256) + AES-CTR con IV aleatorio por mensaje + firma ECDSA. Este esquema se integra como una transformación del payload en la capa de aplicación, sin alterar el comportamiento del modelo físico del canal ni la configuración de la interfaz UAN. En la recepción se aplica la política operativa *verify → decrypt*, es decir, la autenticidad e integridad del mensaje se validan antes de ejecutar el descifrado. Este enfoque permite evitar gasto computacional sobre mensajes inválidos y distinguir experimentalmente entre pérdidas atribuibles al canal y descartes derivados de la política de seguridad.

El impacto del esquema seguro sobre el sistema puede explicarse por dos mecanismos principales. Por un lado, el incremento del tamaño del paquete debido al encapsulado criptográfico, que introduce campos adicionales como IV, clave pública y firma digital. Por otro lado, el costo de procesamiento asociado al cifrado y firmado en el emisor, así como a la verificación y descifrado en el receptor. Ambas dimensiones se analizan posteriormente en las métricas de desempeño.

4.1.1 Condiciones experimentales (6 escenarios)

La evaluación considera seis escenarios que representan el cruce entre el perfil de seguridad y el modelo de amenaza. En todos los casos se conserva la misma arquitectura general (gateway + N nodos) y la misma instrumentación de medición; lo que cambia es el comportamiento criptográfico y/o la presencia/configuración del nodo atacante.

Perfil sin seguridad (NS).

En NS, los nodos transmiten mensajes sin encapsulación criptográfica (sin cifrado y sin

firma). Este perfil sirve como línea base para cuantificar el desempeño de la red sin overhead computacional ni aumento de tamaño de tramas asociado a seguridad.

Perfil con seguridad (S).

En S se activa un esquema híbrido (a nivel de aplicación) que agrega operaciones de seguridad para confidencialidad e integridad/autenticación. En términos operacionales, este perfil introduce dos efectos esperados sobre el desempeño:

- Overhead de comunicación: los paquetes aumentan de tamaño por la inclusión de campos de seguridad (por ejemplo, material criptográfico, IV, firma, etc.).
- Overhead computacional: el emisor y el receptor realizan operaciones criptográficas (cifrado/descifrado y verificación) que agregan tiempo de procesamiento y pueden afectar la latencia.

Modelo de atacante y modos de operación.

Se consideran dos comportamientos del atacante en el nodo atacante (atknode), parametrizados por su aplicación de ataque y su modo:

- Atacante activo: intenta perturbar la red mediante generación/inyección de tráfico (por ejemplo, envío periódico de tramas para elevar colisiones/ocupación del medio).
- Atacante pasivo (listener): se configura en modo escucha (promiscuous mode) para capturar tráfico sin transmitir activamente durante la ejecución.

Escenarios implementados

1. NS_Normal: red sin seguridad y sin atacante (baseline).
2. NS_Atacante: red sin seguridad con atacante en modo atacante activo.
3. S_Normal: red con seguridad habilitada, sin atacante.
4. S_Atacante: red con seguridad habilitada y atacante en modo atacante activo.

5. NS_Listener: red sin seguridad con atacante en modo listener (escucha pasiva).
6. S_Listener: red con seguridad y atacante en modo listener.

En el archivo de configuración final del escenario T4 (t4.ini) se habilita explícitamente el cifrado para el gateway en S_Normal, S_Atacante y S_Listener, alineándolo con el modo seguro de los sensores. Esto evita el caso de recepción a nivel MAC pero rechazo/no aceptación a nivel aplicación (p.ej., goodput = 0 y "DATA OK" = 0) que puede ocurrir cuando emisor y receptor no están configurados de forma consistente. En esta versión actualizada, S_Listener representa un escenario listener pasivo con seguridad end-to-end.

Los módulos de aplicación involucrados en la implementación son *FullApp* (nodos sensores), *Gwapp* (gateway) y *Atackapp* (atacante). El mensaje operativo de intercambio se define en *AppNodePacket.msg*, donde se encapsulan tanto los campos funcionales como los campos criptográficos cuando el perfil seguro está activado. La campaña de ejecución se organiza a través de archivos INI, que permiten conmutar entre perfiles mediante parámetros como numberOfNodes, sim-time-limit y el flag cifrar.

Tabla 38

Resumen de escenarios y perfiles ejecutados (campaña final).

Escenario	**Seguridad**	**Atacante**	**Propósito de evaluación**
NS_Normal	No	Ninguno	Línea base de desempeño sin mecanismos criptográficos.

NS_Atacante	No	Activo (Atackapp)	Cuantificar degradación por intervención/inyección sin protección.
S_Normal	Sí (ECDH+KDF+AES-CTR+ECDSA)	Ninguno	Costo y beneficio de seguridad en operación normal.
S_Atacante	Sí (ECDH+KDF+AES-CTR+ECDSA)	Activo (Atackapp)	Robustez del esquema frente a atacante activo y costo adicional.
NS_Listener	No	Pasivo (listener)	Exposición del tráfico en claro ante escucha del medio (sin inyección).
S_Listener	Sí (ECDH+KDF+AES-CTR+ECDSA)	Pasivo (listener)	Verificar confidencialidad frente a escucha (payload cifrado).

Densidad	**#Nodos**	**Propósito**	**Escenarios**	**Métricas objetivo**
D20	20	Carga moderada	6 escenarios	Delay E2E, PDR/PLR, throughput/goodput, overhead

				(+robustez según escenario)
D50	50	Carga alta	6 escenarios	Delay E2E, PDR/PLR, throughput/goodput, overhead (+robustez según escenario)
D100	100	Estrés de red	6 escenarios	Delay E2E, PDR/PLR, throughput/goodput, overhead (+robustez según escenario)

Nota. Los escenarios se ejecutan sobre la misma pila base y el mismo modelo de canal descritos en el Capítulo 3; las diferencias se activan por parámetros en el archivo INI de campaña.

Parámetros INI relevantes para trazabilidad. La conmutación entre perfiles se realiza vía parámetros de configuración, destacando: numberOfNodes, sim-time-limit y el flag cifrar para habilitar/deshabilitar seguridad. En perfil S, además de cifrar, se genera firma ECDSA y se adjuntan iv/publicKey/signature.

La evaluación se apoya en una pila base de comunicación subacuática ya modelada (medio acústico e interfaz UAN), que permite representar un canal de alta latencia y baja tasa.

En esta tesis, dichos componentes se mantienen constantes para preservar validez comparativa: los seis escenarios se diferencian por la activación del encapsulado seguro y/o por el modo del atacante, pero no por cambios del canal. Con ello, las variaciones en delay, PDR/PLR, throughput/goodput y overhead se atribuyen al factor experimental (seguridad/atacante) y no a cambios en el modelo físico. En términos de alcance, la caracterización del canal queda restringida a lo que el medio simulado modela directamente, y cualquier fenómeno no parametrizado se reporta como limitación del entorno.

4.1.2 Barrido de densidad

Las densidades seleccionadas representan tres niveles de carga progresiva: D20 (carga moderada), D50 (carga alta) y D100 (condición de estrés). El objetivo no es únicamente observar desempeño absoluto, sino identificar tendencias de escalabilidad, es decir, determinar en qué punto el overhead criptográfico se vuelve significativo, cuándo domina la contención del medio y cómo se amplifica el impacto del atacante bajo congestión.

Para cada combinación de densidad y escenario se ejecutaron tres repeticiones con semillas distintas, totalizando 54 corridas experimentales. Este diseño mantiene comparabilidad directa entre densidades y escenarios, evitando variaciones metodológicas entre condiciones.

Se ejecuta un barrido por densidad con D20, D50 y D100 (N = {20, 50, 100} nodos sensores).

Para controlar el tiempo total de simulación sin degradar el valor estadístico, se definió:

Para N = 20, 50 y 100 → n = 3 repeticiones por condición.

Esto da un total de 54 simulaciones:

(3 densidades × 6 escenarios × 3 repeticiones) = 54.

Tabla 39

Matriz final de ejecución (densidad × escenario × repeticiones)

Densidad (N)	NS_Normal	NS_Atacante	S_Normal	S_Atacante	NS_Listener	S_Listener
20	3	3	3	3	3	3
50	3	3	3	3	3	3
100	3	3	3	3	3	3

Nota. Total de ejecuciones = 6 escenarios × 3 densidades × 3 repeticiones = 54 corridas. Este diseño permite comparaciones "controladas" por densidad (misma N) y también análisis de tendencia al incrementar N dentro del mismo escenario.

En todos los casos se ejecutan los seis escenarios por densidad con n=3 repeticiones, totalizando 54 corridas, lo que permite reportar agregados por condición y evitar que las conclusiones no dependan de un único tamaño de red.

4.1.3 Horizonte temporal de simulación

El horizonte recomendado para asegurar una ventana de datos suficiente y permitir convergencia operativa es:

- sim-time-limit = 8000 s como valor base.
- En caso de que S_Atacante con N=100 no produzca ventana estable suficiente (o no converja), se habilita un ajuste a 10000 s únicamente para esa condición, reportándolo explícitamente para trazabilidad.

El horizonte temporal se define para garantizar una ventana de observación suficiente que incluya (i) la convergencia operativa (JOIN), (ii) la fase estable de transmisión de datos, y (iii) condiciones bajo amenaza cuando aplique. En caso de que alguna combinación (por ejemplo, S_Atacante con N=100) no produzca una ventana estable, el ajuste del tiempo de simulación se reporta explícitamente como parte de la trazabilidad experimental.

4.1.4 Parámetros controlados y trazabilidad

Para garantizar que la comparación sea atribuible a los factores experimentales (seguridad/atacante/densidad), se mantienen controlados:

- Modelo de canal acústico y NIC (invariantes respecto a Capítulo 3).
- Topología/área de despliegue y parámetros físicos del medio.
- Tasa nominal de interfaz y temporización base del escenario.
- Misma instrumentación de métricas para todas las condiciones.

Las diferencias entre condiciones se activan por parámetros de configuración (p. ej. activación de cifrar=true/false y modo del atacante listener=true/false, y/o presencia del atacante únicamente en las condiciones que lo requieren). La trazabilidad de cada corrida se preserva mediante los atributos estándar de OMNeT++ en los archivos de resultados (configuración, repetición, semilla, tiempo de simulación, etc.).

Para garantizar comparaciones justas, las variables controladas incluyen el modelo de canal acústico y NIC, la topología/área de despliegue, la temporización base, y la misma instrumentación en todas las condiciones. Las variables manipuladas se restringen a: densidad (N) y escenario (NS/S con normal/atacante/listener). La trazabilidad se preserva porque cada corrida queda identificada por configuración, repetición y semilla en los archivos estándar de OMNeT++ (.sca/.vec), permitiendo reconstruir tablas y gráficas por condición. Podemos observar a continuación las siguientes 3 tablas las cuales nos dan los resultados capturados.

Tabla 40

N = 20 Resultados globales

Escenario	T_proc_nodos (s)	Delay_JOIN (s)	T_convergencia (s) [lim 8000]	Converge (frac)	PktSize_nodo_tx (B)	PktSize_GW_tx (B)	Throughput_GW_rx (bps)	Goodput_GW_rx (bps)	Eficiencia G/T	PDR MAC unicast	DATA OK (#)
NS_Normal	0.668	4.980	75.2	1.00	25.9	10.0	0.203	0.071	0.350	0.998	3959
S_Normal	0.589	9.575	158.2	1.00	265.9	170.0	2.543	0.061	0.024	0.980	1675
NS_Listener	0.668	4.980	75.2	1.00	25.9	10.0	0.203	0.071	0.350	0.998	3959
S_Listener	—	8.211	134.4	1.00	244.0	10.0	96.934	0.000	0.000	0.536	0
NS_Atacante	0.694	4.876	7978.3	1.00	20.2	10.0	0.180	0.060	0.332	0.708	246
S_Atacante	—	10.562	8000.0	0.00	244.0	170.0	1.816	0.000	0.000	0.095	0

Tabla 41

N = 50 Resultados globales

Escenario	T_proc_nodos (s)	Delay_JOIN (s)	T_convergencia (s) [lim 8000]	Converge (frac)	PktSize_nodo_tx (B)	PktSize_GW_tx (B)	Throughput_GW_rx (bps)	Goodput_GW_rx (bps)	Eficiencia G/T	PDR MAC unicast	DATA OK (#)
NS_Normal	0.668	4.980	75.2	1.00	25.9	10.0	0.203	0.071	0.350	0.998	3959
S_Normal	0.589	9.575	158.2	1.00	265.9	170.0	2.543	0.061	0.024	0.980	1675
NS_Listener	0.668	4.980	75.2	1.00	25.9	10.0	0.203	0.071	0.350	0.998	3959
S_Listener	—	8.211	134.4	1.00	244.0	10.0	96.934	0.000	0.000	0.536	0
NS_Atacante	0.694	4.876	7978.3	1.00	20.2	10.0	0.180	0.060	0.332	0.708	246
S_Atacante	—	10.562	8000.0	0.00	244.0	170.0	1.816	0.000	0.000	0.095	0

Tabla 42

N = 100 Resultados globales

Escenario	T_proc_nodos (s)	Delay_JOIN (s)	T_convergencia (s) [cap 8000]	Converge (frac)	PktSize_nodo_tx (B)	PktSize_GW_tx (B)	Throughput_GW_rx (bps)	Goodput_GW_rx (bps)	Eficiencia G/T	PDR MAC unicast	DATA OK (#)
NS_Normal	0.662	5.025	201.9	1.00	25.1	10.0	0.210	0.057	0.272	0.989	3896
S_Normal	0.584	12.755	2237.6	1.00	250.3	170.0	2.438	0.027	0.011	0.398	1255
NS_Listener	0.662	5.025	201.9	1.00	25.1	10.0	0.210	0.057	0.272	0.989	3896
S_Listener	—	13.480	2657.9	1.00	244.0	10.0	6.611	0.000	0.000	0.159	0
NS_Atacante	0.701	4.946	5791.8	0.67	15.7	10.0	0.188	0.028	0.145	0.349	119
S_Atacante	—	17.446	8000.0	0.00	244.0	170.0	2.049	0.000	0.000	0.056	0

4.2 Validación, tratamiento estadístico y recolección de métricas

Antes de presentar los resultados comparativos (Sección 4.3), se aplicó un procedimiento de validación y consistencia sobre los datos generados por OMNeT++ para asegurar que: (i) las señales y contadores se registraron correctamente, (ii) cada corrida es trazable por escenario–densidad–repetición, y (iii) el tratamiento estadístico es coherente con el comportamiento esperado de una red subacuática (alto retardo, baja tasa y sensibilidad a contención).

La recolección se sustenta en los formatos nativos del simulador: **.sca** para escalares (valores agregados y contadores finales) y **.vec** para señales vectoriales (series temporales). Este enfoque permite una comparación justa porque mantiene idéntica instrumentación en los seis escenarios, variando únicamente los factores experimentales (seguridad/atacante/densidad).

Con el objetivo de asegurar consistencia semántica, se definieron criterios mínimos de aceptación por condición (escenario + densidad). Estos criterios verifican que las variables medidas correspondan a las definiciones operacionales del capítulo y que conserven relaciones esperadas (por ejemplo, PDR en rango válido y tendencias coherentes al aumentar N). La **Tabla 43** resume la evidencia mínima requerida para considerar una condición “lista para análisis”.

Tabla 43

Evidencia mínima de instrumentación (checklist de consistencia).

Criterio de validación	Regla aplicada	Evidencia observada en campaña
Trazabilidad por condición	Cada corrida identificada por *(escenario, N, repetición #0–#2)*	Se ejecutaron 54 corridas: 6 escenarios × 3 densidades × 3 repeticiones
Existencia de archivos por corrida	Deben existir .sca y .vec por corrida	Para cada escenario se registraron 9 .sca y 9 .vec (3 densidades × 3 repeticiones)
Consistencia de densidad	En cada corrida, deben existir métricas para N nodos sensores	En todas las corridas, los escalares de energía se registraron exactamente para N nodos (N = 20, 50, 100)
Rango de PDR	PDR ∈ [0, 1]	En todas las condiciones se obtuvo PDR válido (incluyendo degradación fuerte bajo ataque)
Energía registrada (J y mAh)	Debe existir totalEnergyConsumed y consumed_mAh para nodos y gateway	Se registraron ambos para sensores y gateway en las 54 corridas
Manejo explícito de no-convergencia	Si convergence:last no aparece, se trata como dato censurado	Se aplicó tasa de convergencia y cap temporal (Sección 4.2.2)

Nota. Este checklist garantiza que las tablas comparativas no se construyen sobre datos incompletos, inconsistentes o sin trazabilidad experimental.

En términos de trazabilidad, cada corrida queda identificada por escenario, densidad N y repetición (#0, #1, #2), lo que permite reconstruir tablas agregadas de forma determinista. En particular, para métricas operacionales del JOIN se utiliza convergencia:last del Gateway y delaySignal:last de los nodos, agregando por corrida sobre todos los nodos válidos y luego por repeticiones. Para PDR se usa numReceived en el MAC del Gateway y numSent acumulado en los MAC de los nodos sensores esto se ve en la **Figura 46**. Para atacante activo se utiliza adicionalmente la relación entre numSentBroadcast del atacante y numReceivedBroadcast en el Gateway como indicador del "éxito" del flooding sobre el medio como se ve en la figura **Figura 47**.

Figura 46

Consumo energético total de los nodos en función de la densidad de red bajo diferentes escenarios de seguridad y ataque con 100 nodos

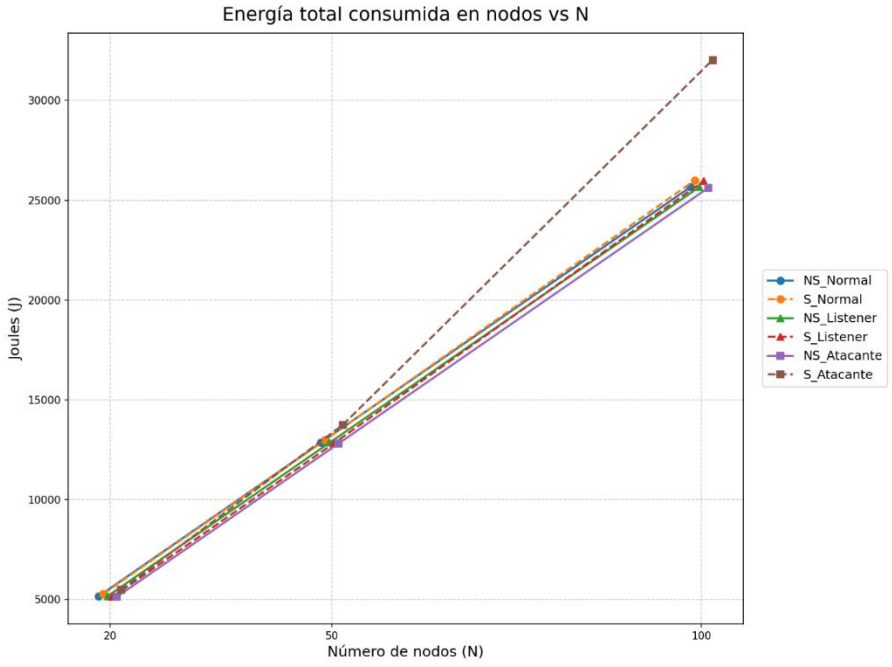
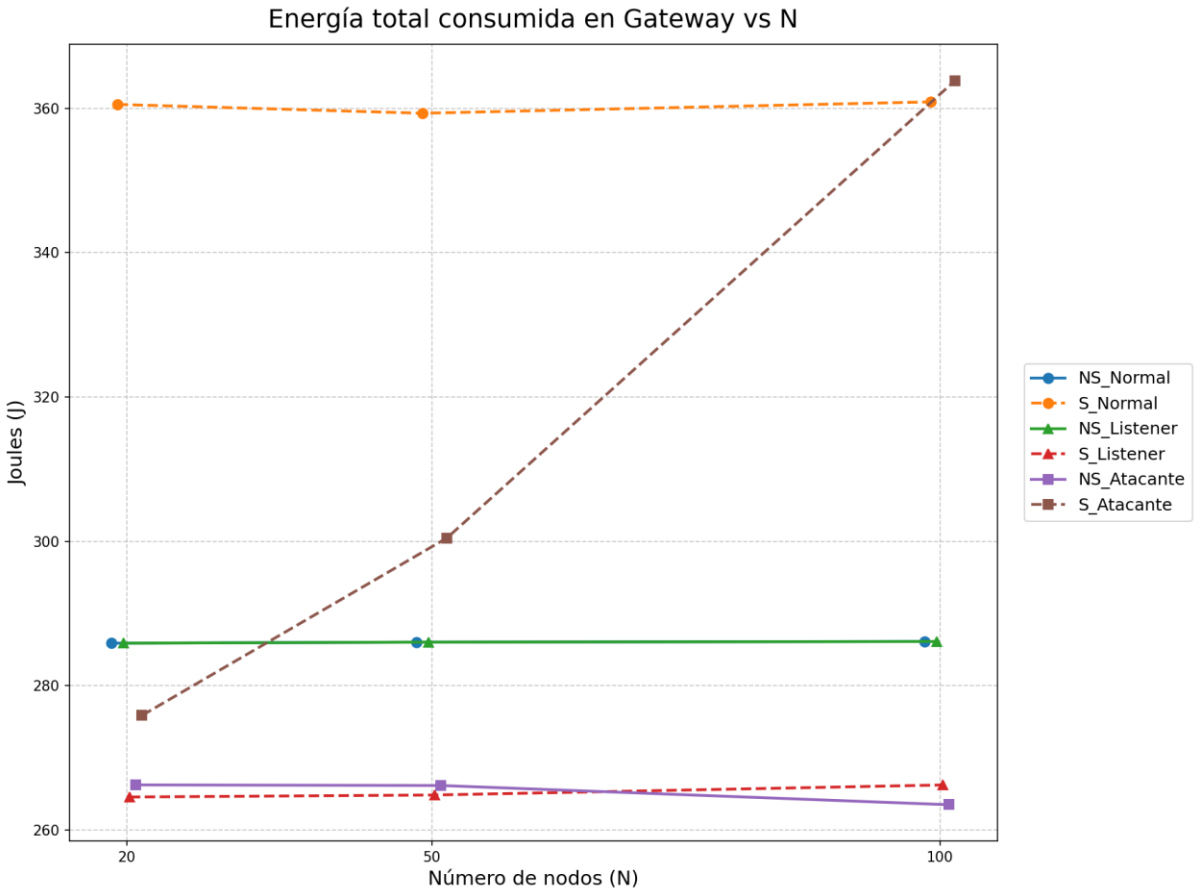


Figura 47

Consumo energético del Gateway en función de la densidad de nodos bajo distintos escenarios de seguridad y ataque



En la **Tabla 44** podemos ver de manera resumida los seis escenarios evaluados y como se estructuran combinando (i) activación o no del mecanismo criptográfico y (ii) tipo de amenaza presente. La Tabla 44 resume la matriz experimental.

Tabla 44

Descripción de escenarios de simulación evaluados.

Código	Seguridad	Condición	Descripción operativa del escenario
NS_Normal	No	Normal	Red operando sin adversario y sin encapsulado criptográfico.
S_Normal	Sí	Normal	Red operando sin adversario con encapsulado/validación criptográfica activada.
NS_Listener	No	Escucha	Se incorpora un nodo adversario pasivo (no transmite). Se evalúa el efecto en desempeño sin seguridad.
S_Listener	Sí	Escucha	Se incorpora escucha pasiva con seguridad activada. Se evalúa impacto en control/flujo y exposición de información.
NS_Atacante	No	Ataque activo	Nodo adversario activo (condición de ataque) sin seguridad activada.
S_Atacante	Sí	Ataque activo	Nodo adversario activo con seguridad activada (se espera mitigación y/o degradación por intento de intrusión/DoS).

Nota. NS = “No Security” sin seguridad; S = “Security” con seguridad.

4.2.1 Fuentes de datos: archivos .sca y .vec

La recolección se basa en los formatos estándar de OMNeT++:

- .sca (scalars): contiene contadores finales y estadísticas agregadas por corrida (por ejemplo: PDR reconstruido desde contadores MAC, tiempo medio de retardo medido sobre eventos JOIN, energía consumida acumulada, etc.).
- .vec (vectors): contiene series temporales por señal (por ejemplo: variación temporal del throughput, jitter, tamaños de paquetes por evento).

Los archivos **.sca** constituyen la base del análisis cuantitativo agregado porque contienen contadores finales y estadísticas resumidas por corrida. En esta campaña se emplean, entre otros, los siguientes elementos: (a) convergencia de JOIN como tiempo total de incorporación de la red; (b) retardo E2E del proceso JOIN capturado por señal; (c) contadores MAC de envío/recepción; (d) descartes (packetDrop*) como evidencia de contención e overhearing; y (e) consumo energético acumulado (totalEnergyConsumed) para sensores y gateway.

Los archivos **.vec**, por su parte, se reservan para el análisis de distribución (histogramas) y comportamiento temporal, especialmente cuando existen colas largas y transitorios que no se describen completamente con promedios.

Archivos .anf (Analysis Files): para la generación de figuras y la trazabilidad de la selección de señales se utilizaron archivos .anf del módulo Analysis de OMNeT++ (por ejemplo: Graficas Escalares_VALIDACION_v2.anf y Graficas Vectores_VALIDACION_v2.anf). Estos archivos almacenan la configuración exacta de los gráficos (filtros por escenario, selección de vectores/escalares, expresiones y reglas de agregación).

En esta campaña, las configuraciones .anf se revisaron para asegurar que: (i) cada métrica se agrupe por condición (escenario, densidad N) y por repetición (#0, #1, #2) sin mezclar corridas; (ii) no existan exclusiones que oculten escenarios o densidades sin declararlo en el texto; y (iii) métricas dependientes del rol, como throughput y goodput, se

extraigan únicamente del Gateway (sumidero). Esta validación reduce el riesgo de errores metodológicos invisibles en las gráficas y fortalece la reproducibilidad del Capítulo 4.

4.2.2 Validación mínima y tratamiento de no-convergencia

En redes subacuáticas, la convergencia del proceso de JOIN puede degradarse por contención, pérdidas y carga adicional de señalización, especialmente al introducir: (i) encapsulado criptográfico y verificación (escenarios S_*), y (ii) un atacante activo que incrementa tráfico o induce fallos operativos (escenarios *_Atacante).

Durante la validación se detectó que, en S_Atacante, la métrica de convergencia no se registra como escalar (convergence:last) en ninguna densidad, lo cual se interpreta como dato censurado por no-convergencia operativa dentro del horizonte de simulación. Para mantener comparabilidad entre condiciones, se reportan dos elementos:

1. **Tasa de convergencia (%)**: fracción de corridas donde el valor existe.
2. **Tiempo de convergencia con cap temporal**: si el valor no existe, se imputa como el límite de simulación (cap = 8000 s), no como convergencia real, sino como indicador conservador de no-cumplimiento.

La **Tabla 45** resume la tasa de convergencia (porcentaje de corridas con convergencia definida) y el tiempo medio de convergencia por escenario y densidad. Esta tabla es clave para interpretar resultados posteriores, ya que si una condición no converge o converge al final del horizonte temporal, métricas de desempeño pueden reflejar un sistema operando fuera de régimen estable.

Tabla 45

Convergencia JOIN por escenario y densidad (n=3 por condición).

Escenario	Nodos	Convergencia (%)	T_convergencia M (s) [lim 8000]	DE (s)
NS_Normal	20	100	75.2	23
NS_Normal	50	100	107.8	26
NS_Normal	100	100	201.9	13.5
S_Normal	20	100	158.2	39.7
S_Normal	50	100	577.8	92.8
S_Normal	100	100	2237.6	254
NS_Listener	20	100	75.2	23
NS_Listener	50	100	107.8	26
NS_Listener	100	100	201.9	13.5
S_Listener	20	100	158.2	39.7
S_Listener	50	100	564.0	111.2
S_Listener	100	100	2197.1	287.2
NS_Atacante	20	100	7978.3	8.1
NS_Atacante	50	100	7957.9	42
NS_Atacante	100	66.7	5791.8	3723.3
S_Atacante	20	0	+8000	0
S_Atacante	50	0	+8000	0
S_Atacante	100	0	+8000	0

Nota. El limite (= 8000 s) representa el horizonte de simulación y se usa únicamente para tratar no-convergencia como dato censurado comparable. En NS_Normal/NS_Listener la convergencia es rápida y estable; al habilitar seguridad (S_Normal) se incrementa

notablemente, especialmente en N = 100, consistente con mayor overhead y presión sobre el canal. En condiciones con atacante, la convergencia se degrada severamente y en S_Atacante no se observa convergencia dentro del horizonte.

En términos interpretativos, los escenarios sin atacante (NS_Normal y NS_Listener) presentan convergencia rápida y estable, con un incremento moderado al aumentar N. Cuando se habilita seguridad (S_Normal y S_Listener) la convergencia se incrementa marcadamente, lo que sugiere un costo adicional asociado al procesamiento y/o a la dinámica de intercambio durante la fase de JOIN. En condiciones con atacante (NS_Atacante) la convergencia se vuelve extremadamente tardía (cercana al horizonte temporal), lo cual es consistente con presión severa sobre el medio debido a inyección de tráfico. En el caso S_Atacante, al no existir registro de convergencia, la condición se maneja como censurada para esta métrica.

4.2.3 Estadísticos reportados y criterios de agregación

Para cada métrica por condición *(escenario, N)* se reporta n = 3, media (M) y desviación estándar (DE). Este criterio permite:

- Controlar variación por semilla (repeticiones independientes).
- Evitar conclusiones basadas en una sola corrida.
- Cuantificar estabilidad (DE baja) versus degradación/aleatoriedad bajo estrés (DE alta).

Adicionalmente, para métricas sensibles a colas largas y comportamiento no estacionario (por ejemplo: retardo de JOIN y su variabilidad), se revisaron percentiles p5 y p95 a partir de señales vectoriales (.vec) en el entorno de análisis. Los percentiles permiten caracterizar dispersión y outliers sin sobreinterpretar el valor medio. En el cuerpo principal se reporta el formato media (DE) por claridad comparativa; los percentiles se utilizaron como verificación de consistencia y soporte para la discusión.

Como ejemplo, se reporta el retardo extremo a extremo (E2E) medido sobre el intercambio JOIN. Esta métrica se obtiene como promedio por corrida y luego se agregan medias y DE entre repeticiones.

Un ejemplo de aplicación de este criterio se observa en el retardo E2E del JOIN. Para cada corrida se calcula la media del retardo observado entre nodos y posteriormente se reportan agregados entre repeticiones. La Tabla 43 resume el retardo medio (por corrida y luego agregado) y su desviación estándar (DE) como indicador de variabilidad entre repeticiones.

Tabla 46

Retardo E2E del JOIN por escenario y densidad (n=3 por condición).

Escenario	**N**	**Delay_JOIN *X* (s)**	**DE (s)**
NS_Normal	20	4.980	0.160
NS_Normal	50	5.058	0.170
NS_Normal	100	5.025	0.161
S_Normal	20	9.575	0.271
S_Normal	50	10.452	0.349
S_Normal	100	12.755	0.319
NS_Listener	20	4.980	0.160
NS_Listener	50	5.058	0.170
NS_Listener	100	5.025	0.161
S_Listener	20	9.591	0.255
S_Listener	50	10.445	0.355
S_Listener	100	12.771	0.804

NS_Atacante	20	4.876	0.030
NS_Atacante	50	4.861	0.112
NS_Atacante	100	4.946	0.126
S_Atacante	20	10.562	1.156
S_Atacante	50	13.415	0.292
S_Atacante	100	17.446	0.735

Nota. El retardo se incrementa al habilitar seguridad (escenarios S_*), consistente con (i) crecimiento del tamaño de paquete por encapsulado y (ii) costo de procesamiento/validación en la ruta Tx/Rx. Bajo condiciones de atacante (especialmente S_Atacante), el retardo aumenta aún más debido a presión adicional del medio y degradación de entrega.

En general, el retardo de JOIN es estable y bajo en condiciones NS sin atacante, mientras que incrementa cuando se habilita seguridad, lo cual es consistente con el costo adicional de verificación y procesamiento criptográfico. En escenarios de alta presión (S_Atacante y S_Listener en N=100), los percentiles altos aumentan significativamente, lo que anticipa la necesidad de complementar promedios con distribuciones (histogramas) en el análisis temporal de 4.3 y 4.4.

4.2.4 Reconstrucción de PDR y consistencia de contadores

La tasa de entrega de paquetes (PDR) se reconstruye utilizando contadores MAC, numReceived en el Gateway y numSent acumulado en los nodos sensores priorizando una medición operativa coherente con el canal subacuático. En términos generales, se compara lo transmitido por los nodos con lo recibido de forma efectiva en el gateway (unicast), manteniendo consistencia de capa y evitando sesgos por tráfico broadcast. La **Tabla 47** presenta la PDR agregada por condición, donde se observa el contraste esperado: alta entrega en condiciones sin ataque y degradación marcada bajo atacante activo.

Tabla 47

PDR por escenario y densidad (n=3 por condición).

Escenario	N	PDR media (%)	Desv. est. (%)	Mín (%)	Máx (%)
NS_Normal	20	99.85	0.03	99.82	99.87
NS_Normal	50	99.71	0.16	99.55	99.88
NS_Normal	100	99.35	0.31	99.05	99.66
NS_Listener	20	99.85	0.03	99.82	99.87
NS_Listener	50	99.71	0.16	99.55	99.88
NS_Listener	100	99.35	0.31	99.05	99.66
NS_Atacante	20	70.79	17.30	51.59	85.16
NS_Atacante	50	66.58	22.26	48.85	91.57
NS_Atacante	100	34.86	7.07	29.83	42.94
S_Normal	20	96.45	0.61	95.84	97.06
S_Normal	50	84.29	1.34	83.09	85.73
S_Normal	100	50.19	5.49	44.75	55.71
S_Listener	20	97.88	0.55	97.37	98.46
S_Listener	50	84.62	1.88	82.59	86.31
S_Listener	100	38.65	3.33	35.36	42.02
S_Atacante	20	9.53	4.07	5.49	13.61
S_Atacante	50	9.09	1.33	7.66	10.34
S_Atacante	100	5.61	0.29	5.32	5.90

Nota. En escenarios sin ataque, NS_Normal/NS_Listener mantienen PDR cercano a 1 incluso al aumentar N. Al habilitar seguridad (S_Normal), PDR disminuye con la densidad, mostrando que el overhead y la contención penalizan la entrega en condiciones de estrés (N = 100). Bajo atacante activo, el deterioro es severo (NS_Atacante y especialmente S_Atacante),

lo cual anticipa condiciones de disponibilidad comprometida y justifica analizar conjuntamente PDR, drops y energía.

Estos resultados permiten establecer consistencia entre escenarios y evidencian que el ataque activo induce un deterioro severo en PDR, mientras que el listener pasivo no introduce degradación por sí mismo. En escenarios con seguridad, la PDR decrece al aumentar N, lo cual sugiere un efecto combinado entre overhead/costo de procesamiento y contención del medio bajo alta densidad. La discusión causal de estas tendencias se desarrolla en 4.4, y su visualización se apoya en gráficos de barras comparativos por densidad (media ± desviación).

4.2.5 Drops y energía como indicadores de presión sobre el medio

Además de PDR, se emplean métricas de descarte (packetDrop*) y consumo energético acumulado como indicadores indirectos de presión sobre el canal y actividad de la NIC. En particular, packetDropNotAddressedToUs captura overhearing y recepción de tramas no dirigidas al nodo, fenómeno que tiende a aumentar con densidad y con presencia de flooding. La **Tabla 48** resume el conteo agregado de este descarte en nodos, evidenciando la relación con el incremento de N y con escenarios atacantes.

Tabla 48

Descartes por "no dirigido a nosotros" en nodos (suma sobre nodos; n=3 por condición).

Escenario	N	DropNotAddr M (#)	DE (#)
NS_Normal	20	151,282	451
NS_Normal	50	392,032	841
NS_Normal	100	795,355	1,224
S_Normal	20	64,716	1,778
S_Normal	50	164,827	2,282

S_Normal	100	401,025	6,388
NS_Listener	20	151,282	451
NS_Listener	50	392,032	841
NS_Listener	100	795,355	1,224
S_Listener	20	60,640	166
S_Listener	50	156,286	1,435
S_Listener	100	316,022	2,243
NS_Atacante	20	15,559	2,870
NS_Atacante	50	39,331	4,302
NS_Atacante	100	88,621	26,370
S_Atacante	20	14,769	4,074
S_Atacante	50	35,321	2,925
S_Atacante	100	72,150	15,836

Nota. Esta métrica captura actividad no útil (overhearing). En condiciones normales sin seguridad, los descartes crecen con N. Al activar seguridad y/o atacante, los patrones cambian debido a variaciones de contención, éxito de recepción y composición del tráfico.

Finalmente, el consumo energético acumulado se reporta en dos unidades: Joules (J) y miliamperio-hora (mAh), ambas registradas por el modelo energético. En términos físicos, ambas son equivalentes bajo el voltaje del sistema, por lo que se presentan separadas para cumplir trazabilidad de resultados y facilitar interpretación práctica (energía vs carga de batería). La **Tabla 49** reporta la energía consumida por sensores (suma sobre nodos) y por el gateway. En general, el consumo escala con N, y las condiciones con mayor presión del medio presentan mayor variabilidad.

Tabla 49

Energía consumida (J) por condición (n=3 por condición).

Escenario	N	E_nodos M (J)	DE_nodos (J)	E_gateway M (J)	DE_gw (J)
NS_Normal	20	5,172.38	0.14	285.90	0.09
NS_Normal	50	12,854.03	0.08	286.04	0.07
NS_Normal	100	25,657.20	0.21	286.14	0.09
S_Normal	20	5,284.07	3.97	360.53	2.92
S_Normal	50	12,986.29	1.12	359.31	1.39
S_Normal	100	25,998.40	3.62	360.88	1.34
NS_Listener	20	5,172.38	0.14	285.90	0.09
NS_Listener	50	12,854.03	0.08	286.04	0.07
NS_Listener	100	25,657.20	0.21	286.14	0.09
S_Listener	20	5,284.43	0.49	353.92	0.27
S_Listener	50	12,988.08	2.49	353.96	0.85
S_Listener	100	25,998.04	39.34	355.83	0.79
NS_Atacante	20	5,128.30	0.89	266.27	0.70
NS_Atacante	50	12,810.46	1.42	266.19	0.15
NS_Atacante	100	25,616.03	2.75	263.52	4.99
S_Atacante	20	5,158.54	3.21	363.90	2.93
S_Atacante	50	12,822.04	1.44	363.82	0.15
S_Atacante	100	32,025.56	437.44	363.84	0.19

Nota. La energía en sensores se reporta como suma sobre N nodos (por eso escala con la densidad). El gateway incrementa su consumo al habilitar seguridad (S_Normal y S_Atacante) debido a mayor carga de recepción y verificación/operación criptográfica **Tabla 50**.

Tabla 50

Consumo de batería (mAh) por condición (n = 3 por condición).

Escenario	N	Consumo_nodos M (mAh)	DE (mAh)	Consumo_gw M (mAh)	DE_gw (mAh)
NS_Normal	20	435.385	0.012	24.066	0.007
NS_Normal	50	1,081.989	0.006	24.078	0.006
NS_Normal	100	2,159.697	0.018	24.086	0.008
S_Normal	20	444.787	0.334	30.348	0.246
S_Normal	50	1,093.122	0.094	30.245	0.117
S_Normal	100	2,188.417	0.304	30.377	0.113
NS_Listener	20	435.385	0.012	24.066	0.007
NS_Listener	50	1,081.989	0.006	24.078	0.006
NS_Listener	100	2,159.697	0.018	24.086	0.008
S_Listener	20	444.818	0.041	29.791	0.023
S_Listener	50	1,093.272	0.209	29.795	0.072
S_Listener	100	2,188.387	3.311	29.952	0.066
NS_Atacante	20	431.675	0.075	22.413	0.059
NS_Atacante	50	1,078.321	0.119	22.406	0.013
NS_Atacante	100	2,156.232	0.232	22.182	0.420
S_Atacante	20	434.223	0.270	30.655	0.246
S_Atacante	50	1,079.295	0.121	30.649	0.013
S_Atacante	100	2,695.755	36.839	30.627	0.016

Nota. En el modelo energético de la simulación, J y mAh son consistentes bajo el voltaje del sistema. Puede verificarse la equivalencia con:

$$E(J) = V \cdot (mAh \cdot 3.6) \quad (7)$$

(por ejemplo, 24.066 mAh ≈ 285.90 J para V ≈ 3.3 V). Se presentan ambas unidades porque OMNeT++ entrega ambos escalares y en tesis es útil interpretarlo como energía total (J) o como carga de batería (mAh).

Con lo anterior, los escenarios quedan validados para reporte comparativo: existe trazabilidad por condición, se controlan censuras de convergencia sin "inventar" convergencia, y se reportan estadísticos consistentes por densidad y escenario. En la Sección 4.3 se presentan los resultados por métrica (tiempo de procesamiento, tamaños de paquete, throughput/goodput, etc.) y en 4.4 se discute el trade-off seguridad–desempeño.

4.3 Resultados por métrica

En esta sección se presentan y comparan los resultados obtenidos para las seis condiciones de evaluación: NS_Normal, NS_Listener, NS_Atacante, S_Normal, S_Listener y S_Atacante, considerando tres densidades de red (D20, D50 y D100, equivalentes a 20, 50 y 100 nodos sensores, respectivamente).

Las métricas analizadas corresponden a las definidas en la sección metodológica (convergencia del proceso JOIN, retardo E2E del JOIN, overhead por encapsulado, PDR/PLR, throughput/goodput en el gateway y coste criptográfico por tiempo de procesamiento). Para cada combinación escenario–densidad, los resultados se resumen como promedios agregados de las repeticiones (n = 3 por condición, salvo donde se indique lo contrario).

Cuando una métrica puede presentar no convergencia, se utiliza el criterio de censura a 8000 s (tiempo límite de simulación) para mantener comparabilidad en los escenarios donde el proceso no finaliza dentro de la ventana simulada.

En la Sección 4.3, los resultados se resumen mediante estadística descriptiva e inferencial calculada sobre tres repeticiones independientes por combinación de escenario y densidad ($n = 3$). La media se denota como μy la desviación estándar muestral como σ; a partir de esta última se obtiene la varianza como $\mathrm{Var} = \sigma^2$. Para cuantificar la incertidumbre de la estimación de μ, se reporta además el intervalo de confianza al 95% ($IC_{95\%}$), calculado mediante la distribución t de Student:

Ecuación 7

Distribución t de Student

$$IC_{95\%} = \mu \pm t_{0.025,2}\left(\frac{\sigma}{\sqrt{3}}\right) \quad (7)$$

donde $t_{0.025,2} = 4.303$

corresponde a $n - 1 = 2°$

En consecuencia, el margen del intervalo se expresa como $2.484 \cdot \sigma$. Con el fin de mantener las tablas compactas, cada celda presenta el resumen estadístico evitando listar explícitamente los valores individuales de cada repetición.

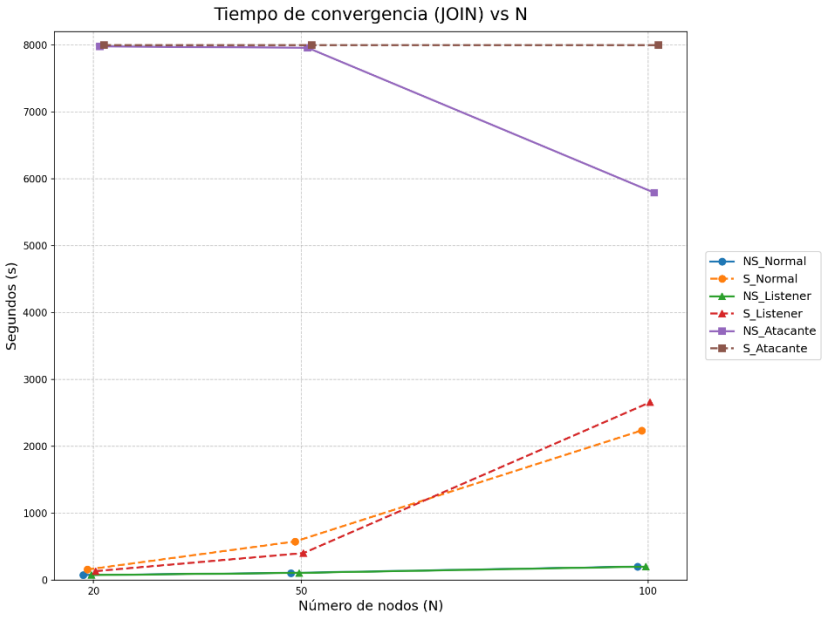
4.3.1 Convergencia JOIN

La convergencia JOIN representa el tiempo necesario para que el proceso de incorporación y sincronización de la red alcance un estado estable (i.e., cuando el último nodo logra completar el ciclo de JOIN). Esta métrica permite evaluar escalabilidad y robustez del mecanismo de acceso inicial ante variaciones de densidad y presencia de atacante.

La **Figura 48** presenta el tiempo de convergencia del proceso JOIN por escenario en función de la densidad (D20, D50 y D100). Cuando no se alcanza convergencia dentro de la ventana de simulación, el valor se censura a 8000 s para mantener comparabilidad.

Figura 48

Tiempo de convergencia JOIN (lim = 8000 s) en función de la densidad, por escenario.



Nota. El valor 8000 s representa censura por no convergencia dentro del tiempo máximo de simulación.

En D20, los escenarios NS_Normal/NS_Listener presentan convergencia rápida (≈ 75 s), evidenciando que, sin mecanismos criptográficos, el proceso de incorporación es estable a baja densidad. En contraste, el escenario S_Normal incrementa la convergencia (≈ 158 s), lo cual es coherente con el aumento de mensajes y tamaño de carga útil que introduce la seguridad. El escenario NS_Atacante muestra convergencia cercana al límite temporal (≈ 7978 s), lo que indica que la interferencia del atacante degrada el proceso de ingreso aun sin seguridad.

En D50, el patrón se acentúa: NS_Normal/NS_Listener mantienen convergencia baja (≈ 108 s), mientras que S_Normal incrementa a ≈ 578 s, reflejando mayor ocupación del medio y reintentos. El escenario NS_Atacante permanece cercano al límite (≈ 7958 s). En esta densidad, aunque el atacante no necesariamente impide todo JOIN en NS, sí lo retrasa hasta el borde de la ventana de simulación.

En D100, el efecto de densidad es decisivo: NS_Normal/NS_Listener suben a ≈ 202 s (todavía operable), mientras S_Normal escala de forma marcada hasta ≈ 2238 s, mostrando que la seguridad impacta fuertemente la escalabilidad del JOIN en alta densidad. Para NS_Atacante, además del aumento de tiempo, aparece pérdida de convergencia (fracción de convergencia ≈ 0.67), indicando que parte de las repeticiones no completaron JOIN. Finalmente, en S_Atacante se observa no convergencia (censura al límite), lo que caracteriza este escenario como no operativo en la fase de incorporación.

La **Tabla 51** muestra el retardo promedio de JOIN (media y desviación estándar) por escenario y densidad. En NS (sin seguridad) el retardo se mantiene estable alrededor de ~5 s, con variaciones pequeñas al incrementar N. Al habilitar seguridad (S) el retardo aumenta sistemáticamente, lo cual es consistente con el costo adicional de procesamiento/verificación y mayor carga por encapsulado.

Tabla 51

Retardo de JOIN (s) por escenario y densidad (DE)

Escenario	**20**	**50**	**100**
NS_Normal	4.98 (0.16)	5.06 (0.17)	5.03 (0.16)
S_Normal	9.57 (0.27)	10.45 (0.35)	12.75 (0.76)
NS_Listener	4.98 (0.16)	5.06 (0.17)	5.03 (0.16)
S_Listener	9.59 (0.26)	10.45 (0.36)	12.77 (0.80)
NS_Atacante	4.88 (0.03)	4.86 (0.11)	4.95 (0.13)

S_Atacante	10.56 (0.82)	13.42 (0.81)	17.45 (1.14)

Nota. Valores en formato media (DE). La DE se interpreta como estimador de jitter (variabilidad del retardo). El retardo JOIN se expresa en segundos.

El costo de seguridad en JOIN crece con N, especialmente bajo condiciones de presión (S_Normal pasa de ~9.6 s a ~12.8 s).

S_Atacante exhibe los mayores retardos (hasta ~17.45 s) y alta dispersión, coherente con mayor contención/tráfico no útil o fallas de establecimiento estable.

En términos de incorporación, la seguridad incrementa el tiempo de convergencia y su efecto crece con la densidad. Ante atacante activo, la convergencia puede volverse no alcanzable en la ventana temporal, lo que evidencia que el mecanismo de seguridad (por sí solo) no garantiza disponibilidad frente a saturación del medio.

4.3.2 Retardo E2E (medido sobre el intercambio JOIN)

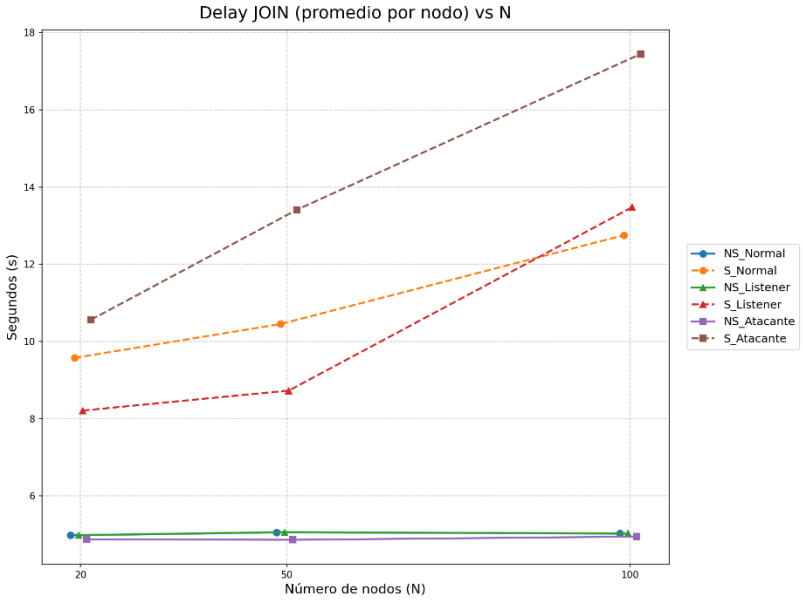
El retardo E2E del JOIN estima el tiempo de extremo a extremo asociado al intercambio de incorporación (desde transmisión del mensaje JOIN en el nodo hasta su recepción/gestión en el gateway dentro del flujo del protocolo). Esta métrica captura el impacto del medio acústico y de la carga de control sobre la latencia inicial.

Además del valor medio del retardo, se reporta la variabilidad temporal de la incorporación (jitter) como la desviación estándar (DE) del retardo E2E del JOIN, calculada por corrida y agregada por condición (n = 3). Un jitter elevado indica mayor dispersión en los tiempos de entrega, típicamente asociada a contención del medio, reintentos y colas en el canal acústico. Cuando la métrica proviene de señales vectoriales (.vec), la interpretación se complementa con la inspección de percentiles p5 y p95 para caracterizar colas largas y valores atípicos sin depender solo del promedio.

Aclaración sobre jitter: en esta tesis el término se emplea como indicador de variabilidad del retardo (packet delay variation) y se operacionaliza mediante la dispersión del retardo E2E (DE). No se asume sincronización global perfecta ni un modelo de señal continua; por ello, el jitter se interpreta en términos de eventos de envío/recepción registrados por OMNeT++, esto es visto en la **Figura 49**.

Figura 49

Retardo promedio de convergencia JOIN por nodo en función de la densidad de red bajo distintos escenarios de seguridad y ataque.



En D20, los escenarios NS mantienen retardo estable alrededor de ≈ 4.98 s, mientras S_Normal aumenta a ≈ 9.58 s, es decir, cerca del doble. El incremento es consistente con el mayor tamaño de los paquetes y el procesamiento asociado al esquema seguro. En S_Listener se registra ≈ 8.21 s, mostrando que la condición con seguridad mantiene latencias elevadas incluso sin que el adversario sea el factor dominante. En S_Atacante el retardo crece más (≈ 10.56 s), lo cual coincide con mayor congestión y reintentos.

En D50, el retardo NS permanece cercano a ≈ 5.06 s, mientras S_Normal se ubica alrededor de ≈ 10.45 s. La condición S_Atacante incrementa hasta ≈ 13.42 s, reforzando el efecto de adversario sobre la latencia.

En D100, se mantiene estabilidad relativa en NS (≈ 5.03 s), pero el retardo crece en S_Normal (≈ 12.75 s) y se maximiza bajo adversario en S_Atacante (≈ 17.45 s). El aumento de densidad intensifica la cola de espera y la probabilidad de retransmisión, por lo que el esquema seguro se ve más afectado a medida que incrementa el número de nodos. En la **Tabla 52** se mira a mayor detalle.

La convergencia se analiza con dos indicadores:

- Tiempo de convergencia (s) con censura (lim 8000 s)
- Porcentaje de corridas convergentes (sobre n = 3 por condición)

Tabla 52

Tiempo de convergencia (s) y porcentaje de convergencia por escenario y densidad (lim = 8000 s; n = 3)

Escenario	**20**	**50**	**100**
NS_Normal	75.2 / 100%	107.8 / 100%	201.9 / 100%
S_Normal	158.2 / 100%	577.8 / 100%	2237.6 / 100%
NS_Listener	75.2 / 100%	107.8 / 100%	201.9 / 100%
S_Listener	158.2 / 100%	564.0 / 100%	2197.1 / 100%
NS_Atacante	7978.3 / 100%	7957.9 / 100%	5791.8 / 67%
S_Atacante	8000.0 / 0%	8000.0 / 0%	8000.0 / 0%

Nota. Cada celda presenta Tiempo (s) / % corridas convergentes. El tiempo está censurado a 8000 s.

4.3.3 Overhead de encapsulado (tamaño de paquete)

El overhead de encapsulado se evalúa mediante el tamaño medio de paquete transmitido por los nodos sensores (TX) y por el gateway (TX), ya que el esquema de seguridad agrega campos (p. ej., material criptográfico, autenticación y/o metadatos) que elevan la carga útil efectiva. En un canal de 1200 bps, el tamaño de paquete es crítico porque incrementa el tiempo de ocupación del medio, reduce oportunidades de acceso y eleva la probabilidad de colisiones.

El overhead criptográfico se evidencia directamente en el tamaño promedio de paquetes transmitidos. En NS el nodo transmite ~25 B y el gateway ~10 B (tráfico muy compacto). En S, el nodo incrementa el tamaño de forma notable (≈250–266 B) por la inclusión de material criptográfico (campos de seguridad). Se detalla a mayor rasgo en la **Tabla 53**.

Tabla 53

Tamaño promedio de paquete transmitido (B) por nodo y gateway (n = 3)

(a) Nodo TX (B)

Escenario	**20**	**50**	**100**
NS_Normal	25.9	25.5	25.1
S_Normal	265.9	262.2	250.3
NS_Listener	25.9	25.5	25.1
S_Listener	244.0	244.0	244.0
NS_Atacante	20.2	17.9	15.7
S_Atacante	244.0	244.0	244.0

(b) Gateway TX (B)

Escenario	**20**	**50**	**100**

NS_Normal	10.0	10.0	10.0
S_Normal	170.0	170.0	170.0
NS_Listener	10.0	10.0	10.0
S_Listener	10.0	10.0	10.0
NS_Atacante	10.0	10.0	10.0
S_Atacante	170.0	170.0	170.0

Nota. Tamaños promedio en bytes (B). Se reporta el promedio agregado de muestras de TX.

Los resultados de overhead se presentan en la **Tabla 54**, se resumen los tamaños promedio observados. En términos generales, NS_Normal/NS_Listener mantienen tamaños de nodo cercanos a ≈ 25 B, mientras que S_Normal sube a ≈ 250–266 B, es decir, aproximadamente 10×. A modo ilustrativo, a 1200 bps un paquete de ~25 B (≈ 200 bits) requiere del orden de 0.17 s de transmisión ideal, mientras que ~266 B (≈ 2128 bits) requiere ~1.77 s, sin contar cabeceras MAC/PHY, guard times ni retransmisiones. Esta diferencia explica por qué el esquema seguro impacta la convergencia, el retardo y el PDR cuando crece la densidad.

Tabla 54

Tamaño de paquete (TX) por escenario y densidad (n = 3 por condición).

Escenario	**N**	**PktSize nodo TX (B) (M)**	**PktSize nodo TX (B) (DE)**	**PktSize GW TX (B) (M)**	**PktSize GW TX (B) (DE)**
NS_Normal	20	25.86	0.01	10.00	0.00
NS_Normal	50	25.46	0.03	10.00	0.00
NS_Normal	100	25.13	0.02	10.00	0.00

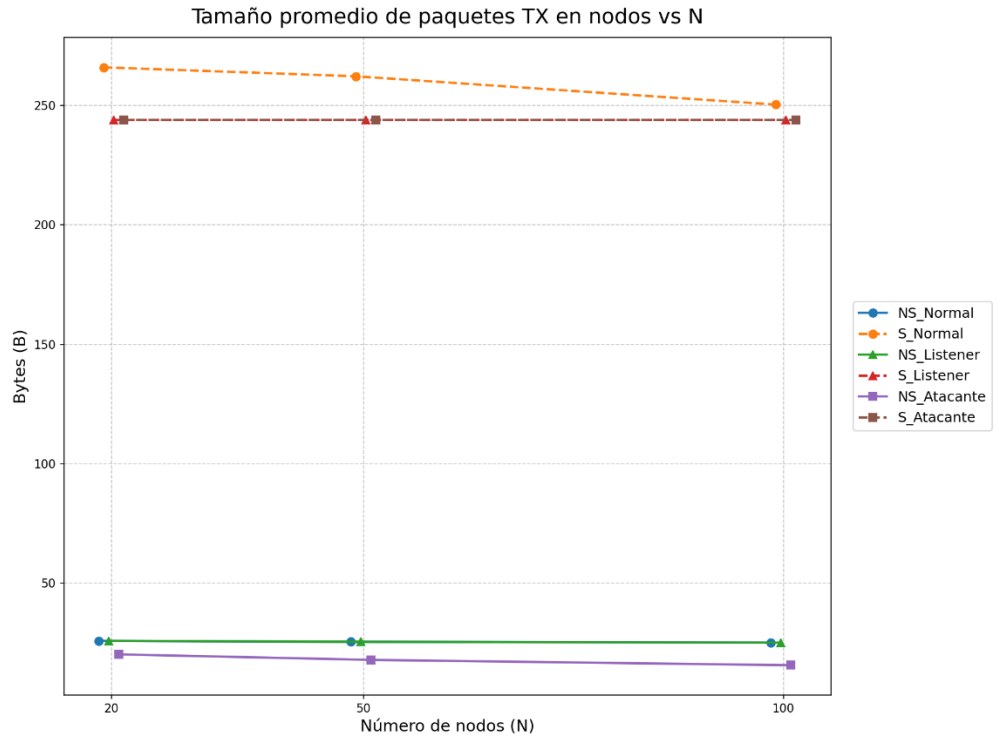
NS_Listener	20	25.86	0.01	10.00	0.00
NS_Listener	50	25.46	0.03	10.00	0.00
NS_Listener	100	25.13	0.02	10.00	0.00
NS_Atacante	20	20.20	1.86	10.00	0.00
NS_Atacante	50	17.89	0.46	10.00	0.00
NS_Atacante	100	15.66	1.60	10.00	0.00
S_Normal	20	265.92	0.20	170.00	0.00
S_Normal	50	262.18	0.53	170.00	0.00
S_Normal	100	250.33	0.84	170.00	0.00
S_Listener	20	285.37	0.37	170.07	0.06
S_Listener	50	278.26	0.99	170.00	0.00
S_Listener	100	255.99	1.57	170.00	0.00
S_Atacante	20	244.00	0.00	170.00	0.00
S_Atacante	50	244.00	0.00	170.00	0.00
S_Atacante	100	244.00	0.00	170.00	0.00

Nota. M = media; DE = desviación estándar entre repeticiones. Los tamaños se reportan en bytes.

La **Figura 50** resume el tamaño medio de paquete transmitido por el nodo sensor (TX) en función de la densidad, destacando el overhead introducido por el encapsulado seguro.

Figura 50

Tamaño medio de paquete transmitido por el nodo sensor en función de la densidad, por escenario.



En condiciones con atacante, el tamaño de nodo puede mantenerse alto (por presencia de tramas del flujo seguro), pero el efecto dominante se observa en la ocupación del canal y en la degradación de las métricas de entrega, más que en una reducción del tamaño nominal.

4.3.4 PDR/PLR (tasa de entrega y pérdida)

El PDR (Packet Delivery Ratio) mide la fracción de paquetes entregados exitosamente (a nivel MAC unicast) respecto a los transmitidos, siendo un indicador directo de confiabilidad. A continuación, en la **Tabla 55** esta métrica permite identificar degradación por congestión, colisiones, interferencia y/o ataques.

Tabla 55

PDR MAC unicast (%) por escenario y densidad (media y DE; n = 3)

Escenario	**20**	**50**	**100**
NS_Normal	99.85 (0.03)	99.71 (0.16)	98.90 (0.23)
S_Normal	97.98 (0.55)	85.26 (1.86)	39.76 (3.48)
NS_Listener	99.85 (0.03)	99.71 (0.16)	98.90 (0.23)

S_Listener	97.88 (0.55)	84.62 (1.88)	38.65 (3.33)
NS_Atacante	70.79 (17.30)	66.58 (22.26)	34.86 (7.07)
S_Atacante	9.53 (4.07)	9.09 (1.33)	5.61 (0.29)

Nota. Valores en formato media (DE) expresados en porcentaje.

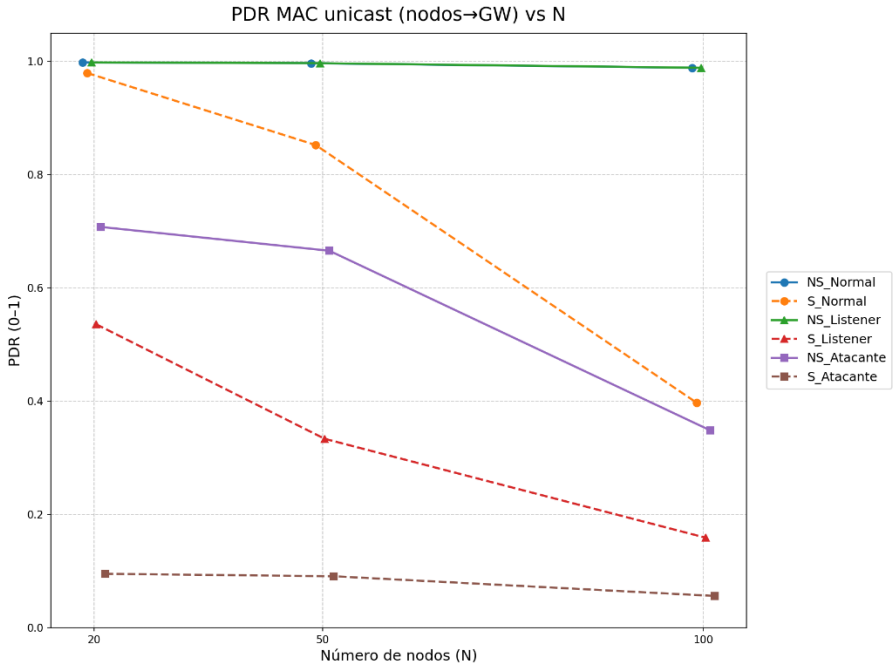
En condiciones NS_Normal/NS_Listener, el PDR se mantiene alto en todas las densidades (≈ 0.998, 0.997 y 0.989 para D20, D50 y D100), lo que evidencia estabilidad del tráfico no seguro incluso al incrementar nodos. Sin embargo, en S_Normal el PDR disminuye conforme crece la densidad: alrededor de 0.980 en D20, 0.853 en D50 y 0.398 en D100. Esto confirma que el esquema seguro es más sensible al aumento de nodos debido a su overhead y mayor ocupación del canal.

En presencia de atacante, NS_Atacante reduce el PDR de forma progresiva conforme aumenta N (≈ 0.708, 0.666, 0.349). En el caso seguro, S_Atacante presenta valores críticos (≈ 0.095, 0.091, 0.056) y además coincide con escenarios donde no se logra convergencia JOIN, por lo que el sistema se considera severamente degradado en disponibilidad y entrega.

La **Figura 51** muestra el PDR (MAC unicast) por escenario y densidad, permitiendo visualizar la degradación por congestión, overhead criptográfico y presencia de atacante.

Figura 51

PDR (MAC unicast) en función de la densidad, por escenario.



El PDR confirma que sin seguridad la red escala mejor en confiabilidad; mientras que con seguridad, la confiabilidad decrece a medida que aumenta la densidad. El atacante agrava esta degradación, especialmente en el perfil seguro, donde se observa colapso operativo (bajo PDR y no convergencia).

4.3.5 Throughput y Goodput

Aclaración metodológica (Gateway): throughput y goodput se reportan exclusivamente en el Gateway (Boya) por ser el sumidero común y punto de comparación entre condiciones. No se reporta goodput en los nodos sensores, ya que estos actúan como emisores y no como destino final del tráfico; por tanto, una "tasa útil" a nivel de sensor podría confundirse con tasa de envío o retransmisiones. En consecuencia, el análisis filtra explícitamente el módulo del Gateway en las configuraciones de extracción (archivos .anf), evitando mezclar mediciones entre roles.

El throughput en el gateway representa los bits recibidos por unidad de tiempo (tráfico total recibido). El goodput representa los bits útiles de aplicación efectivamente aceptados/procesados en el gateway (carga útil válida). Esta distinción es clave: un sistema

puede mostrar throughput alto por congestión (muchos bits llegando), pero goodput bajo si esos datos no son útiles o no son aceptados por el protocolo.

Los valores se obtienen a partir de las señales troughput y goodput (registradas en el gateway), y se consolidan por corrida y luego por repeticiones, evitando inflar el tamaño muestral. La **Tabla 56** resume throughput, goodput y la eficiencia G/T. En condiciones NS_Normal/NS_Listener, el throughput promedio se ubica alrededor de 0.20–0.23 bps, con goodput en torno a 0.057–0.071 bps, generando eficiencias ≈ 0.27–0.35. En contraste, S_Normal incrementa el throughput a ≈ 2.44–2.56 bps, pero el goodput se mantiene en magnitudes similares o menores (≈ 0.027–0.061 bps), por lo que la eficiencia cae a ≈ 0.011–0.024. Esto evidencia que, aunque llegan más bits al gateway en el esquema seguro (por overhead), la cantidad de información útil no aumenta proporcionalmente.

Tabla 56

Throughput, goodput y eficiencia (G/T) en el gateway (n = 3 por condición) en perfiles seguros.

Escenario	N	Throughput GW RX (bps) (M)	Throughput GW RX (bps) (DE)	Goodput GW RX (bps) (M)	Goodput GW RX (bps) (DE)	Eficiencia G/T (M)	Eficiencia G/T (DE)
NS_Normal	20	0.2026	0.0052	0.0710	0.0040	0.3503	0.0115
NS_Normal	50	0.2303	0.0037	0.0662	0.0027	0.2874	0.0075
NS_Normal	100	0.2096	0.0020	0.0571	0.0010	0.2724	0.0025
NS_Listener	20	0.2026	0.0052	0.0710	0.0040	0.3503	0.0115
NS_Listener	50	0.2303	0.0037	0.0662	0.0027	0.2874	0.0075
NS_Listener	100	0.2096	0.0020	0.0571	0.0010	0.2724	0.0025

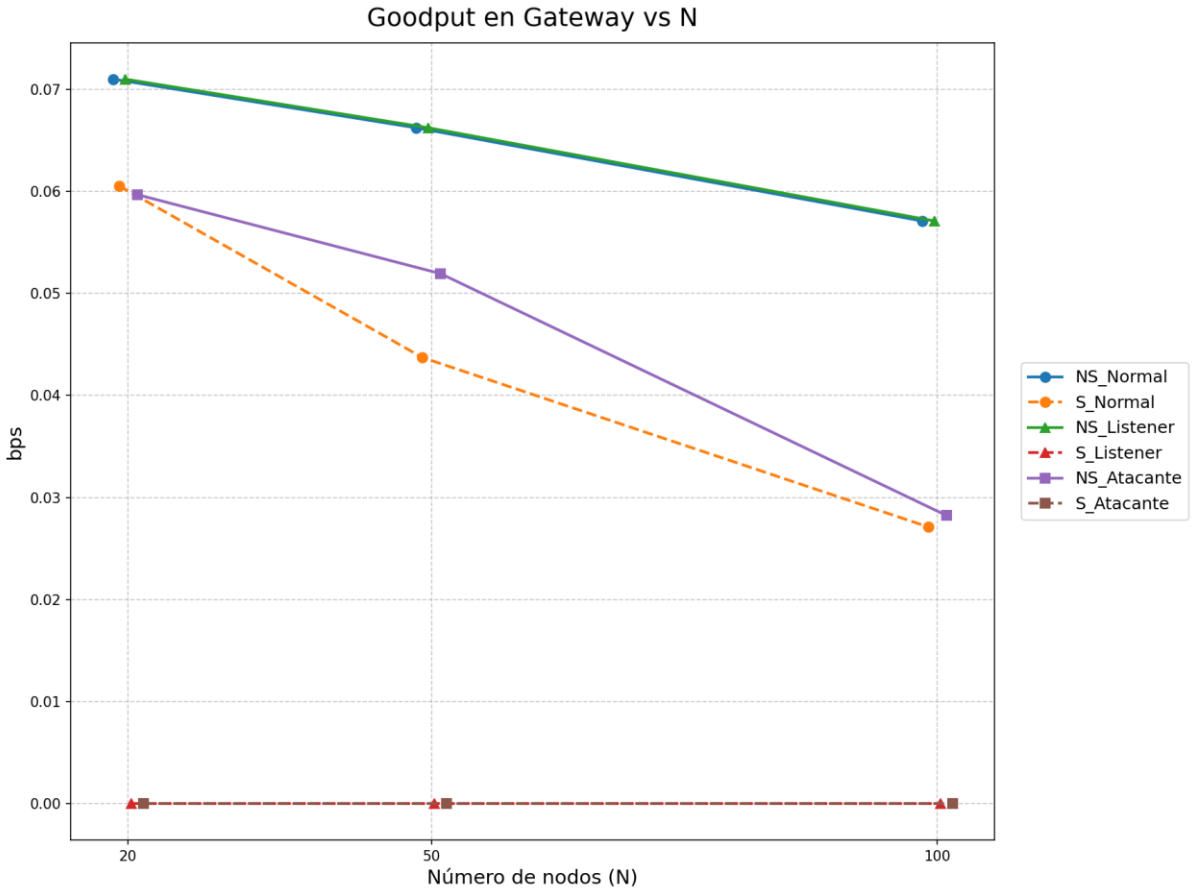
NS_Atacante	20	0.1796	0.0065	0.0597	0.0085	0.3316	0.0349
NS_Atacante	50	0.1973	0.0025	0.0519	0.0024	0.2631	0.0095
NS_Atacante	100	0.1878	0.0130	0.0282	0.0245	0.1446	0.1255
S_Normal	20	2.5433	0.0795	0.0605	0.0033	0.0238	0.0013
S_Normal	50	2.5608	0.0573	0.0437	0.0029	0.0171	0.0014
S_Normal	100	2.4377	0.0930	0.0271	0.0010	0.0111	0.0001
S_Listener	20	2.7229	0.1199	0.1027	0.0057	0.0378	0.0023
S_Listener	50	2.7107	0.0517	0.0740	0.0049	0.0273	0.0021
S_Listener	100	2.5638	0.1089	0.0469	0.0017	0.0183	0.0003
S_Atacante	20	1.8164	0.0426	0.0000	0.0000	0.0000	0.0000
S_Atacante	50	1.8548	0.0789	0.0000	0.0000	0.0000	0.0000
S_Atacante	100	2.0494	0.3106	0.0000	0.0000	0.0000	0.0000

Nota. M = media; DE = desviación estándar entre repeticiones. Goodput = 0 indica ausencia de carga útil válida recibida/procesada en el gateway.

La **Figura 52** compara el goodput medido en el gateway por escenario y densidad, enfatizando la diferencia entre tráfico recibido y datos útiles aceptados a nivel de aplicación.

Figura 52

Goodput en el gateway en función de la densidad, por escenario.



Con seguridad, el sistema incrementa el tráfico total (throughput) principalmente por overhead, pero reduce la eficiencia útil (goodput/throughput). En escenarios adversarios, el goodput puede colapsar a cero, mostrando que el esquema aporta control de aceptación (autenticidad/integridad), pero la red sigue siendo vulnerable a pérdida de disponibilidad por congestión/saturación.

4.3.6 Coste criptográfico (Tiempo Procesamiento)

El coste criptográfico se aproxima mediante el **tiempo de procesamiento en el nodo** asociado a la preparación del primer flujo de datos (incluyendo generación/encapsulado del mensaje y operaciones necesarias según el perfil). Esta métrica permite contrastar si la penalización del esquema seguro proviene del cómputo o si el efecto dominante es la comunicación, en la **Tabla 57** se puede ver a más detalle.

Tabla 57

Tiempo de procesamiento promedio en nodos (s) por escenario y densidad (media y DE; n = 3)

Escenario	20	50	100
NS_Normal	0.668 (0.022)	0.658 (0.008)	0.662 (0.004)
S_Normal	0.589 (0.025)	0.589 (0.006)	0.584 (0.024)
NS_Listener	0.668 (0.022)	0.658 (0.008)	0.662 (0.004)
S_Listener	0.579 (0.031)	0.587 (0.004)	0.586 (0.014)
NS_Atacante	0.694 (0.062)	0.724 (0.046)	0.701 (0.002)
S_Atacante	—	—	—

Nota. "—" indica ausencia de muestras/medición aplicable en esa condición.

La **Tabla 58** presenta los valores agregados. En escenarios NS_Normal/NS_Listener se observan tiempos alrededor de 0.658–0.668 s. Para S_Normal, los tiempos se mantienen en ≈ 0.584–0.589 s. Aunque el esquema seguro introduce operaciones criptográficas, los resultados muestran que el tiempo registrado no se incrementa de forma dominante frente al no seguro; por tanto, en esta configuración el desempeño queda más condicionado por el canal acústico de baja tasa y el overhead de transmisión que por el cómputo local.

En S_Atacante no se registran muestras (—) debido a que el sistema no alcanza fase operativa de datos válida (goodput nulo), por lo que el indicador no es aplicable en esa condición.

Tabla 58

Tiempo de procesamiento en nodos (s) por escenario y densidad.

Escenario	N	T_proc_nodos (s) (M)	T_proc_nodos (s) (DE)	n (muestras)

NS_Normal	20	0.668	0.022	3
NS_Normal	50	0.658	0.008	3
NS_Normal	100	0.662	0.004	3
NS_Listener	20	0.668	0.022	3
NS_Listener	50	0.658	0.008	3
NS_Listener	100	0.662	0.004	3
NS_Atacante	20	0.694	0.062	3
NS_Atacante	50	0.724	0.046	3
NS_Atacante	100	0.701	0.002	2
S_Normal	20	0.589	0.025	3
S_Normal	50	0.589	0.006	3
S_Normal	100	0.584	0.024	3
S_Listener	20	0.579	0.031	3
S_Listener	50	0.587	0.004	3
S_Listener	100	0.586	0.014	3
S_Atacante	20	—	—	0
S_Atacante	50	—	—	0
S_Atacante	100	—	—	0

Nota. "—" indica métrica no registrada/no aplicable por ausencia de fase de datos válida. En NS_Atacante con N=100 se dispone de n=2 por ausencia de registro en una repetición.

El coste criptográfico medido en tiempo no es el componente que explica la degradación global; el principal factor es el overhead de tamaño de paquete y la ocupación del canal a 1200 bps, que se traduce en mayor retardo, menor PDR y pérdida de convergencia cuando crece la densidad o aparece un adversario.

A partir de las métricas, se observa un patrón consistente: el esquema con seguridad incrementa el tamaño del paquete alrededor de un orden de magnitud y, por tanto, eleva la ocupación del canal. Como resultado, S_Normal es operativo en densidades bajas y medias (D20–D50), pero su desempeño se degrada con fuerza en D100 (aumento de convergencia JOIN y caída de PDR). Frente a adversario activo (S_Atacante), el sistema prioriza control de aceptación (goodput nulo por ausencia de datos válidos), pero pierde disponibilidad (no convergencia y PDR crítico).

4.3.7 Comparación directa: overhead de seguridad (S_Normal vs NS_Normal)

Para cuantificar el costo/impacto del perfil de seguridad en operación normal, en la **Tabla 59** se presenta el cambio porcentual relativo entre **S_Normal** y **NS_Normal**, por densidad.

Tabla 59

Overhead relativo de seguridad en operación normal: Δ% (S_Normal vs NS_Normal) por densidad

N	ΔProc_%	Delay_%	Convergencia_%	PktSize Nodo_%	PktSize GW_%	Throughput GW_%	GoodputGW_%	Eficiencia_%	PDR_pp	DATA_OK_%
20	-11.90	92.28	110.2	928.47	1600	1155.4	-14.72	-93.20	-1.87	-57.68

50	-10.47	106.63	435.95	929.71	1600	1011.77	-33.98	-94.05	-14.45	-60.08
100	-11.81	153.83	1007.99	896.25	1600	1062.82	-52.49	-95.91	-59.14	-67.78

Nota. ΔPDR_pp representa la diferencia en **puntos porcentuales**. Throughput y goodput corresponden al gateway.

- La seguridad incrementa drásticamente el tráfico total (throughput) por overhead, pero **reduce la eficiencia** (hasta ~-96%).
- Al escalar a N=100, el sistema con seguridad en operación normal sufre una caída fuerte en PDR (≈ -59 pp) y en DATA OK (~-68%), mostrando un límite práctico de escalabilidad bajo las condiciones actuales del canal y encapsulado.

4.3.8 Energía consumida

Como indicador operacional adicional, se reporta la energía total consumida por nodos (suma) y por gateway. Esta métrica ayuda a argumentar costo energético asociado a mayor actividad de transmisión/recepción y presión del medio visto en la **Tabla 60** *Energía consumida (J) por escenario y densidad (media y DE; n = 3)*.

Tabla 60

Energía consumida (J) por escenario y densidad (media y DE; n = 3)

Escenario	20 Nodos	20 GW	50 Nodos	50 GW	100 Nodos	100 GW
NS_Normal	5,172.38 (0.14)	285.90 (0.09)	12,854.03 (0.08)	286.04 (0.07)	25,657.20 (0.21)	286.14 (0.09)
S_Normal	5,284.07 (3.97)	360.53 (2.92)	12,986.29 (1.12)	359.31 (1.39)	25,998.40 (37.23)	360.88 (0.48)
NS_Listener	5,172.38 (0.14)	285.90 (0.09)	12,854.03 (0.08)	286.04 (0.07)	25,657.20 (0.21)	286.14 (0.09)
S_Listener	5,124.34 (1.34)	264.59 (0.06)	12,828.05 (2.02)	264.87 (0.10)	25,967.70 (119.39)	266.26 (0.12)
NS_Atacante	5,128.30 (0.89)	266.27 (0.70)	12,810.46 (1.42)	266.19 (0.15)	25,616.03 (2.75)	263.52 (4.99)
S_Atacante	5,479.58 (47.29)	275.90 (4.09)	13,751.37 (337.70)	300.48 (9.77)	32,025.56 (6,457.82)	363.84 (74.65)

Nota. Valores en formato media (DE) en joules (J).

4.3.9 Tasa de error de bit (BER)

La tasa de error de bit (*Bit Error Rate*, BER) es una medida de confiabilidad del enlace que expresa la proporción de bits que se reciben con error respecto al total de bits transmitidos. En términos generales, se define como:

Ecuación 8

Ecuación de Tasa de error de bit (BER)

$$BER = N_{err}/N_{tot} \quad 8$$

Donde,

N_{err}es el número de bits erróneos

N_{tot}el número total de bits transmitidos/recibidos (según el punto de medición).

En redes de sensores subacuáticos (UWSN) esta métrica es especialmente relevante debido a la naturaleza del canal acústico (alta atenuación, multitrayecto, ruido y variabilidad), que puede degradar la confiabilidad de la transmisión a nivel físico. En este contexto, no se dispone de un conteo explícito y estable de N_{err}por paquete como para reportar un BER físico directo consistente en todos los escenarios. En particular, cuando un paquete alcanza la capa superior se contabiliza como "correcto", y la degradación observada en densidades altas o bajo ataque se manifiesta predominantemente como pérdida de paquetes (disponibilidad), más que como "paquetes recibidos con bits corruptos".

Para atender el requerimiento de reportar BER y mantener trazabilidad con los datos efectivamente medidos, se emplea un BER equivalente (BER_{eq}). Este indicador transforma la pérdida de paquetes observada en la probabilidad por bit que produciría esa misma pérdida, bajo el supuesto clásico de errores independientes por bit. Es importante precisar que BER_{eq}no pretende reemplazar un BER físico estimado por demodulación/codificación; su propósito es expresar la confiabilidad observada (a través de PLR/PDR) en una escala por bit que permita comparación entre escenarios.

Para atender el requerimiento del indicador, se estima un BER equivalente (BER_{eq})a partir de la pérdida observada y la longitud promedio del paquete. Primero se obtiene $PDR = R_x/T_x$y $PLR = 1 - PDR$. Bajo el supuesto clásico de errores independientes por bit, la probabilidad de recepción correcta de un paquete de $\bar{L}$bits es $(1-BER)^{\bar{L}}$. Aproximando $1 - PLR \approx (1 - BER)^{\bar{L}}$, se despeja:

Ecuación 9

Tasa de error de bit Equivalente BER_{eq}

$$BER_{eq} = 1 - (1 - PLR)^{\frac{1}{L}} \quad 9$$

Los resultados (Tabla 61) muestran BER_{eq}muy bajo en NS_Normal/NS_Listener (orden 10^{-7}–10^{-6}), y un incremento marcado en NS_Atacante (hasta 10^{-3}en $N = 100$), consistente con el aumento de pérdidas bajo ataque activo. En los escenarios seguros, BER_{eq}crece con la densidad por mayor presión del canal asociada al overhead criptográfico.

Tabla 61

BER equivalente promedio BER_{eq} *por escenario y densidad*

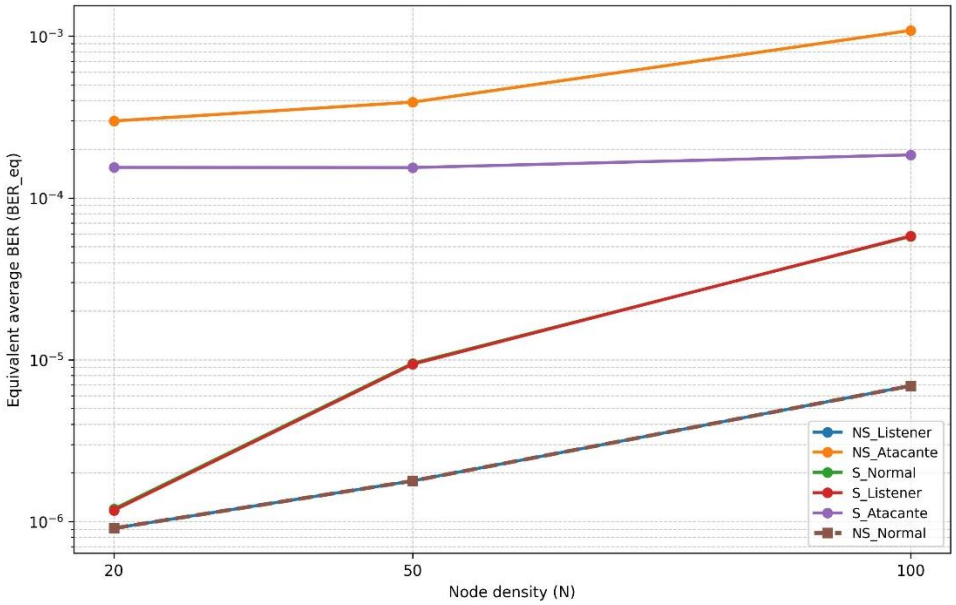
Escenario	**N=20**	**N=50**	**N=100**
NS_Normal	9.0995×10^{-7}	1.7856×10^{-6}	6.9026×10^{-6}
NS_Listener	9.0995×10^{-7}	1.7856×10^{-6}	6.9026×10^{-6}
NS_Atacante	2.9886×10^{-4}	3.9058×10^{-4}	1.0834×10^{-3}
S_Normal	1.1977×10^{-6}	9.5124×10^{-6}	5.7736×10^{-5}
S_Listener	1.1748×10^{-6}	9.3895×10^{-6}	5.8189×10^{-5}
S_Atacante	1.5423×10^{-4}	1.5399×10^{-4}	1.8441×10^{-4}

Nota. BER_{eq} se calcula desde $PLR = 1 - PDR$ y la longitud promedio del paquete $\bar{L}$(bits) mediante $BER_{eq} = 1 - (1 - PLR)^{1/\bar{L}}$. Los valores corresponden al promedio de tres repeticiones por combinación escenario–densidad y representan una equivalencia por bit basada en pérdida de paquetes, no un conteo físico directo de bits erróneos.

La siguiente **Figura 53** presenta el comportamiento del BER equivalente promedio (BER_eq) en función de la densidad de nodos (N = 20, 50 y 100) para los seis escenarios evaluados: NS_Normal, NS_Listener, NS_Atacante, S_Normal, S_Listener y S_Atacante. La métrica BER_eq permite cuantificar el nivel efectivo de errores de transmisión considerando tanto las condiciones del canal acústico como el impacto de los mecanismos de seguridad y la presencia de atacantes.

Figura 53

Comportamiento del BER equivalente BER_{eq} en función de la densidad de nodos para los distintos escenarios de seguridad y ataque



El análisis gráfico se muestra en escala logarítmica, lo que facilita la comparación entre órdenes de magnitud y permite observar con mayor claridad las variaciones entre

escenarios seguros (S) y no seguros (NS), así como el efecto del incremento de densidad sobre la degradación del canal. Esta representación resulta clave para interpretar la robustez del esquema criptográfico implementado frente a condiciones de mayor congestión y escenarios adversariales.

4.4 Discusión

Los resultados obtenidos permiten interpretar el impacto del esquema híbrido ECIES-based y de los modelos de amenaza sobre el desempeño de la UWSN bajo condiciones comparables, dado que el canal acústico, la topología y la pila de comunicación se mantuvieron constantes entre perfiles. La comparabilidad se refuerza mediante trazabilidad por condición (escenario, densidad N, repetición) y validación de la extracción de métricas desde fuentes .sca/.vec.

El efecto directo de la seguridad se evidencia en el proceso de incorporación (JOIN): al incrementarse el tamaño de los mensajes y añadirse operaciones de firma/verificación, el retardo medio aumenta y, simultáneamente, se incrementa la variabilidad del retardo (jitter) en escenarios de mayor contención. En este marco, un jitter elevado se interpreta como inestabilidad temporal (dispersión) del proceso de entrega asociada a reintentos, colas y competencia por el medio acústico.

En términos de eficiencia, la diferencia entre throughput y goodput cuantifica el costo operativo del overhead: el throughput en el Gateway refleja el tráfico total recibido, mientras que el goodput representa la carga útil de aplicación efectivamente aceptada. Al habilitar seguridad, el throughput puede incrementarse por el aumento de bytes por paquete; sin embargo, la eficiencia útil (G/T) disminuye, especialmente en densidades altas, donde el canal de 1200 bps se convierte en el principal cuello de botella.

El análisis de los escenarios con atacante activo confirma que, bajo el modelo utilizado, el ataque se manifiesta principalmente como presión sobre el canal (flooding). En

este régimen, la disponibilidad del medio se vuelve el factor dominante: aun cuando el esquema criptográfico preserva confidencialidad e integridad del payload y aplica la política verify → decrypt para evitar descifrar mensajes no auténticos, la saturación del canal limita la entrega efectiva y provoca colapso del goodput en el Gateway. Por ello, la criptografía debe complementarse con mecanismos de disponibilidad (control de tráfico, mitigación en MAC o detección/aislamiento).

Los escenarios con Listener validan un comportamiento esperado en redes compartidas: un atacante pasivo puede observar el medio sin introducir degradación apreciable del desempeño. En perfiles NS, esta condición expone el payload; en perfiles S, el contenido permanece cifrado y autenticado, por lo que la escucha no compromete la confidencialidad ni la integridad.

Respecto a escalabilidad, la densidad actúa como amplificador: en densidades bajas (N=20) el sistema opera con estabilidad relativa; al aumentar N, el overhead criptográfico incrementa la ocupación del canal y degrada progresivamente métricas de control y entrega (retardo, jitter y PDR). En consecuencia, el rango de densidad moderada emerge como un punto de operación más equilibrado para el perfil seguro bajo los parámetros del escenario.

Los indicadores complementarios de drops y energía apoyan la interpretación causal: el incremento de descartes y consumo energético bajo seguridad y/o ataque sugiere mayor overhearing, reintentos y permanencia activa del radio. Esta evidencia es coherente con la hipótesis de que la degradación global está dominada por congestión del medio y no únicamente por el costo computacional del cifrado.

En síntesis, la campaña experimental evidencia un trade-off claro: el esquema híbrido implementado aporta confidencialidad, integridad y autenticación (verify → decrypt), pero introduce costos observables en tamaño de paquete, retardo y eficiencia útil, con impacto creciente a medida que aumenta la densidad o se reduce la disponibilidad del canal. Estos

resultados deben interpretarse dentro del alcance del modelo (simulación por eventos discretos), considerando no-convergencia como dato censurado y evitando extrapolaciones a métricas no instrumentadas a nivel de señal física.

4.5 Amenazas y robustez

Esta sección discute las amenazas a la validez y la robustez de los resultados presentados, considerando que la evaluación se realiza en un entorno simulado con componentes estocásticos y con un modelo de atacante deliberadamente acotado. El objetivo no es eliminar la incertidumbre inherente a la simulación, sino explicitarla y describir las medidas adoptadas para minimizar sesgos, preservar trazabilidad y sostener conclusiones reproducibles.

En términos de validez interna, la variabilidad entre corridas proviene principalmente de mecanismos estocásticos del simulador, tales como temporización de eventos, contención del medio y decisiones dependientes de la semilla aleatoria. Para mitigar el sesgo asociado a esta variabilidad, el diseño experimental incorpora repeticiones por condición y densidad ($n = 3$ para cada combinación escenario–densidad), y el análisis se ejecuta en dos etapas: primero se agregan métricas por corrida (obteniendo un estadístico representativo por ejecución) y luego se agregan entre repeticiones. Adicionalmente, se controla que cada corrida corresponda exactamente a la condición declarada (perfil NS/S, modo del atacante y densidad N), preservando consistencia entre configuración y análisis. Este control evita confusiones comunes en campañas multi-escenario, como mezclar resultados de densidades distintas o interpretar un perfil con seguridad como si fuera un perfil sin seguridad.

Respecto a la validez de constructo, las métricas se definen de forma operativa a partir de señales y contadores registrados por el simulador. En particular, el retardo E2E se mide sobre el proceso JOIN mediante la señal delaySignal, y la convergencia global se define como el tiempo en que el gateway confirma que todos los nodos completaron JOIN. Para métricas

basadas en archivos .vec, se evita inflar el tamaño muestral agregando primero por corrida (estadístico representativo) y luego por repeticiones. Esta decisión es especialmente relevante cuando existen colas largas o transitorios, ya que mezclar "muestras crudas" entre corridas puede sobreponderar ejecuciones con mayor número de eventos registrados. Asimismo, en perfiles seguros se adopta la política *verify* → *decrypt*, de modo que la aceptación del payload queda condicionada a verificación previa; por ello, una caída de PDR o de goodput puede reflejar defensa activa (rechazo por integridad) y no necesariamente deterioro del canal. Esta distinción se incorpora en la interpretación, evitando atribuir al medio acústico pérdidas que corresponden a la política de seguridad.

Como medida adicional de robustez, las configuraciones de análisis (.anf) utilizadas para generar figuras fueron validadas para garantizar la correcta selección de señales y la agregación por condición (escenario, densidad y repetición), evitando la mezcla involuntaria de corridas o el filtrado accidental de escenarios. Este control metodológico fortalece la trazabilidad entre fuentes (.sca/.vec), tablas y gráficas presentadas en el Capítulo 4.

En cuanto a la validez externa, los resultados se derivan de un modelo de canal acústico simulado y de una pila de comunicación que, aunque representativa para UWSN, permanece simplificada respecto a un despliegue real. Por tanto, la generalización de los hallazgos depende de cuánto se asemejen los parámetros del canal (distancias, propagación, ruido y tasa de bits) a un entorno físico específico. En esta tesis se reportan los parámetros del canal y de la topología en el Capítulo 3 para facilitar replicación y comparación, pero se reconoce que fenómenos no parametrizados en el modelo (por ejemplo, variaciones complejas del entorno, multitrayecto más severo o movilidad no considerada) podrían modificar el comportamiento observado. En consecuencia, las conclusiones deben interpretarse como tendencias bajo condiciones controladas y no como predicciones directas de campo.

La validez de conclusión se respalda mediante el uso de repeticiones por condición (n = 3) y el reporte de estadísticos que incluyen media, desviación estándar y percentiles (p5 y p95) en métricas susceptibles a asimetría y colas largas, como el retardo en escenarios congestionados. El uso de percentiles reduce la dependencia de la media frente a valores extremos y permite caracterizar el comportamiento típico bajo estrés. Además, se aplican reglas explícitas de tratamiento para datos no definidos: cuando una corrida no registra convergencia (por ejemplo, ausencia de la métrica o no finalización dentro del horizonte temporal), el resultado se clasifica como censurado/no convergente y se reporta como tal, evitando reemplazos por valores promedio. Esta decisión incrementa la robustez del análisis al impedir que valores imputados oculten fallas operativas reales.

Finalmente, se explicitan limitaciones del atacante modelado. El "atacante activo" implementado se representa principalmente como un generador de tráfico adicional (flooding) que presiona el medio y aumenta la contención, y no como un atacante con capacidades criptográficas avanzadas, tales como reescritura de mensajes válidos, ataques de tipo Man-in-the-Middle (MITM), compromiso de claves o suplantación basada en firmas. Por ello, las conclusiones de robustez se restringen al efecto de disponibilidad y presión sobre el canal, así como a la reacción del sistema ante tráfico adicional. En el caso del atacante pasivo (listener), la amenaza se interpreta como exposición del contenido en perfiles NS y como verificación de confidencialidad en perfiles S, sin esperar degradación de desempeño por interferencia, dado que no existe inyección. En conjunto, estas delimitaciones permiten interpretar correctamente los resultados: el beneficio principal del esquema propuesto se asocia a confidencialidad e integridad bajo *verify* → *decrypt*, mientras que la degradación bajo atacante activo se explica primordialmente por saturación del medio en un canal acústico de recursos limitados.

Con estas consideraciones, la campaña experimental mantiene trazabilidad, comparabilidad y rigor suficiente para sustentar las conclusiones del capítulo, al tiempo que declara explícitamente los alcances y restricciones de la evaluación bajo simulación.

Conclusiones

La presente investigación demostró que es factible adaptar e integrar un esquema híbrido basado en ECDH + KDF (SHA-256) + AES-CTR con IV aleatorio por mensaje y firma ECDSA dentro de una UWSN simulada en OMNeT++, aplicando una política verify → decrypt en el receptor sin modificar el modelo del canal acústico ni la interfaz UAN.

Los resultados confirmaron un trade-off cuantificable entre seguridad, latencia y eficiencia: al habilitar seguridad, el tamaño de paquete y la ocupación del canal aumentan, elevando el retardo de incorporación (JOIN) y su variabilidad temporal (jitter). Esta degradación se amplifica al incrementar la densidad de nodos, lo que evidencia que, en redes acústicas de baja tasa, el principal cuello de botella operativo es el medio compartido y no únicamente el costo criptográfico.

En presencia de un atacante pasivo (listener), se verificó que la escucha del medio no provoca degradación observable del desempeño, lo cual refuerza la necesidad de cifrado: en perfiles sin seguridad el payload queda expuesto, mientras que en perfiles seguros permanece cifrado y autenticado, preservando confidencialidad e integridad sin penalización adicional por el simple hecho de ser observado.

Bajo atacante activo modelado como inyección de tráfico (flooding), se evidenció que el factor dominante es la disponibilidad del canal. En este régimen, la criptografía protege el contenido y evita descifrar mensajes no auténticos mediante verify → decrypt, pero no impide el colapso de desempeño cuando el medio es saturado. Por ello, se concluye que la seguridad criptográfica debe complementarse con mecanismos de disponibilidad (p. ej., control de tráfico, mitigación en MAC o detección/aislamiento de fuentes anómalas).

Metodológicamente, el estudio se sustenta en métricas extraídas de .sca y .vec con trazabilidad por condición (escenario, densidad N y repetición), y con validación de las configuraciones de análisis (.anf) para evitar mezclas involuntarias de corridas o roles de

medición (por ejemplo: goodput reportado únicamente en el Gateway). Considerando el alcance de simulación por eventos discretos y la censura de no-convergencia, una densidad moderada (aproximadamente hasta 50 nodos) se perfila como un punto de operación más equilibrado para el perfil seguro bajo los parámetros del escenario.

Recomendaciones

Como trabajo futuro, se recomienda ampliar el modelo de simulación para incorporar movilidad realista de los nodos, considerando deriva por corrientes y desplazamientos lentos típicos de despliegues subacuáticos. Esta extensión debería acompañarse de un modelo energético más detallado, que represente explícitamente los estados de la radio (transmisión, recepción, espera e inactividad) y su consumo asociado, además de incluir el costo computacional de las operaciones criptográficas (por ejemplo, generación/verificación de firma y derivación de claves). Con ello, el análisis no solo reflejaría el desempeño en términos de entrega y retardo, sino también el impacto real de la seguridad sobre la autonomía energética, especialmente en redes densas y con tráfico sostenido.

Asimismo, es pertinente extender el modelo de adversario para contemplar amenazas más sofisticadas que no se limiten a degradar disponibilidad por tráfico. En particular, resulta valioso simular ataques man-in-the-middle (MITM), suplantación de identidad/falsificación y replay, debido a que son escenarios que ponen a prueba de forma directa la autenticación e integridad del esquema híbrido propuesto. Para que estos experimentos sean medibles y comparables, conviene instrumentar contadores específicos, como eventos de “rechazo por verificación fallida” o “descartes por autenticación”, de manera que se pueda separar con claridad la pérdida debida al canal (colisiones, saturación) frente a la pérdida causada por seguridad (paquetes descartados por firma inválida). Esto permitiría cuantificar su efecto en métricas como PDR, goodput y retardo, con una interpretación más fina del trade-off seguridad–eficiencia.

En el plano criptográfico, otra recomendación relevante es evaluar curvas elípticas alternativas orientadas a eficiencia, como X25519/Ed25519 (Curve25519), en comparación con secp256r1. Esta comparación debería realizarse bajo el mismo marco experimental para asegurar justicia metodológica, midiendo no solo latencia de cómputo y consumo energético,

sino también el tamaño de mensajes, el overhead total por paquete y el desempeño bajo densidad alta y escenarios con atacante. Esta línea permitiría establecer si una curva más eficiente mejora la viabilidad operativa del esquema en condiciones estrictas de ancho de banda acústico y recursos limitados.

Finalmente, se sugiere explorar la integración del esquema propuesto dentro de enfoques de IoUT (Internet of Underwater Things), orientados a escalabilidad, gestión de servicios y operación cooperativa entre nodos y gateways. En esa dirección, resulta especialmente útil investigar mecanismos ligeros de gestión y distribución de claves adaptados a canales de alta latencia y baja tasa, así como opciones de trazabilidad que no impongan cargas excesivas (por ejemplo, estructuras tipo *ledger* liviano o registros resumidos) para auditoría mínima de intercambios, sin comprometer el rendimiento. Esta evolución permitiría transitar desde una evaluación puntual de seguridad en comunicación hacia un marco más completo de operación segura a nivel de sistema subacuático conectado.

Trabajos futuros

Como trabajo futuro inmediato, se recomienda extender la evaluación incorporando movilidad y variabilidad ambiental, dado que el canal subacuático real presenta dinámicas temporales más complejas que las capturadas por un modelo estacionario. La inclusión de movilidad (por ejemplo, desplazamientos lentos de nodos y variación gradual de distancias) permitiría analizar la estabilidad de convergencia, la dispersión del retardo y la resiliencia de entrega frente a condiciones más cercanas a un despliegue real.

De manera complementaria, resulta valioso realizar un análisis de sensibilidad del canal variando sistemáticamente parámetros como tasa de bits efectiva, nivel de ruido, alcance de transmisión y pérdida por propagación. Este barrido permitiría identificar regiones de operación donde el overhead criptográfico se vuelve crítico o donde la congestión domina el comportamiento global, aportando criterios más finos para dimensionamiento de redes UWSN seguras.

También se propone fortalecer el modelo de atacante e instrumentación asociada. En particular, sería pertinente incorporar atacantes con estrategias más cercanas a amenazas criptográficas (por ejemplo, suplantación y reenvío) y registrar explícitamente contadores de rechazo por verificación fallida para separar con mayor precisión descartes por integridad frente a pérdidas del canal. Frente a flooding, se recomienda evaluar mecanismos complementarios de disponibilidad, como limitación de tasa (rate limiting), priorización de tráfico legítimo, ventanas de acceso adaptativas o ajustes MAC que reduzcan el impacto del broadcast masivo.

Bibliografía

Ahmed, S., Javaid, N., & Shakeel, M. (2017). A survey on underwater wireless sensor networks: architecture, applications and challenges. *Journal of Network and Computer Applications*, *82*, 24–45. https://doi.org/10.1016/j.jnca.2017.01.015

Ainslie, M. A. (2010). *Principles of sonar performance modelling*. https://doi.org/10.1007/978-3-540-87662-5

Akyildiz, I. F., Pompili, D., & Melodia, T. (2005a). Underwater acoustic sensor networks: Research challenges. *Ad Hoc Networks*, *3*(3), 257–279. https://doi.org/10.1016/j.adhoc.2005.01.004

Akyildiz, I. F., Pompili, D., & Melodia, T. (2005b). Underwater acoustic sensor networks: Research challenges. *Ad Hoc Networks*, *3*(3), 257–279. https://doi.org/10.1016/j.adhoc.2005.01.004

Alharbi, A., & Ibrahim, S. (2026). Adaptive Localization-Free Secure Routing Protocol for Underwater Sensor Networks. *Sensors*, *26*(1), 17. https://doi.org/10.3390/s26010017

Ali, M. F., Jayakody, D. N. K., Chursin, Y. A., Affes, S., & Dmitry, S. (2020). Recent Advances and Future Directions on Underwater Wireless Communications. *Archives of Computational Methods in Engineering*, *27*(5), 1379–1412. https://doi.org/10.1007/s11831-019-09354-8

Ali, M. F., Jayakody, D. N. K., Chursin, Y. A., Affes, S., & Sonkin, D. (2019). Recent Advances and Future Directions on Underwater Wireless Communications. *Archives of Computational Methods in Engineering*, *27*, 1379–1412. https://doi.org/10.1007/s11831-019-09354-8

Ariza, A. (2026). *Basic Undewater propagation model for inet and omnet++, derived from Ns3*. https://github.com/aarizaq/UanModel

Barbeau, M., & Garcia-Alfaro, J. (2015). Security issues in underwater wireless sensor networks. *IEEE Canadian Conference on Electrical and Computer Engineering*. https://doi.org/10.1109/CCECE.2015.7129510

Batool, S. F., & Ahmad, R. (2021). Evaluation of ECIES for Lightweight Secure Communication in Resource-Constrained Devices. *Proceedings of the International Conference on Emerging Trends in Smart Technologies (ICETST)*, 1–5. https://doi.org/10.1109/ICETST52348.2021.9540324

Casari, P., & Zorzi, M. (2022). Protocol design issues in underwater acoustic networks. *IEEE Communications Magazine*, *60*(1), 80–86. https://doi.org/10.1109/MCOM.001.2100123

Chang, S.-H., Pereira, R., & Shih, K.-P. (2015). *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing.* IEEE.

Chang, Y., Chen, Y., & Huang, C. (2015). Underwater optical wireless communications: overview and challenges. *IEEE Communications Magazine*, *53*(11), 84–89. https://doi.org/10.1109/MCOM.2015.7321984

Chaudhary, M., Goyal, N., Benslimane, A., Awasthi, A. K., Alwadain, A., & Singh, A. (2023). Underwater wireless sensor networks: enabling technologies for node deployment and data collection challenges. *IEEE Internet of Things Journal*, *10*(4), 3500–3524. https://doi.org/10.1109/JIOT.2022.3218766

Chitre, M., Shahabudeen, S., & Stojanovic, M. (2008). *Underwater acoustic communications and networking: Recent advances and future challenges*. https://doi.org/10.4031/002533208786861263

Cong, Y., Yang, G., Wei, Z., & Zhou, W. (2010). Security in underwater sensor network. *2010 WRI International Conference on Communications and Mobile Computing, CMC 2010*, *1*, 162–168. https://doi.org/10.1109/CMC.2010.18

Das, S., & Thampi, S. M. (2015). Security issues in underwater wireless sensor networks. *Procedia Computer Science*, *46*, 580–587. https://doi.org/10.1016/j.procs.2015.02.098

Dini, G., & Lo Duca, A. (2012). *A secure communication suite for underwater acoustic sensor networks*. https://doi.org/10.3390/s121115133

Domingo, M. C. (2011). Overview of channel models for underwater wireless communication networks. *Physical Communication*, *1*(3), 163–182. https://doi.org/10.1016/j.phycom.2011.01.001

Dong, H., & Liu, Y. (2011). Localization algorithms for underwater wireless sensor networks. *IEEE Communications Surveys & Tutorials*, *13*(1), 95–106. https://doi.org/10.1109/SURV.2010.092710.00064

European Commission. (2024). *Commission Recommendation (EU) 2024/779 of 26 February 2024 on Secure and Resilient Submarine Cable Infrastructures*. https://eur-lex.europa.eu/eli/reco/2024/779/oj/eng

European Commission. (2025). *Joint Communication to the European Parliament and the Council: EU Action Plan on Cable Security*. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52025JC0009

Felemban, E., Shaikh, F. K., Qureshi, U. M., Sheikh, A., & Qaisar, S. (2015). Underwater Sensor Network Applications: A Comprehensive Survey. *International Journal of Distributed Sensor Networks*. https://doi.org/10.1155/2015/896832

Gallimore, E., Partan, J., Vaughn, I., Singh, S., Shusta, J., & Freitag, L. (2010). *The WHOI Micromodem-2: A Scalable System for Acoustic Communications and Networking*.

Goyal, S. B., Ravi, R. V, Verma, C., Raboaca, M. S., & Enescu, F. M. (2022). *A lightweight cryptographic algorithm for underwater acoustic networks*. https://doi.org/10.1016/j.procs.2022.12.029

Haerbin, E. (2016). *Proceedings of 2016 IEEE International Conference on Electronic Information and Communication Technology : ICEICT 2016 : August 20-22, 2016, Harbin, China*.

Hankerson, D., Menezes, A., & Vanstone, S. (2004). *Guide to Elliptic Curve Cryptography*. Springer. https://doi.org/10.1007/b97644

Heidemann, J., Stojanovic, M., & Zorzi, M. (2012). Underwater sensor networks: applications, advances and challenges. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *370*(1958), 158–175. https://doi.org/10.1098/rsta.2011.0214

Idris, M. Y., & Zukarnain, Z. A. (2020). Performance Analysis of ECIES Scheme in Securing Communication of IoT Devices. *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 1–6. https://doi.org/10.1109/ICCCNT49239.2020.9225615

IEEE Computer Society., IEEE Communications Society., & Institute of Electrical and Electronics Engineers. (2013). *Computers and Communications (ISCC), 2013 IEEE Symposium on : date, 7-10 July 2013.*

International Organization for Standardization, & International Electrotechnical Commission. (2006). *ISO/IEC 18033-2:2006: Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers* (Número ISO/IEC 18033-2:2006). https://www.iso.org/standard/32291.html

Islam, A. A., & Taher, K. A. (2021). A Novel Authentication Mechanism for Securing Underwater Wireless Sensors from Sybil Attack. *2021 5th International Conference on*

Electrical Engineering and Information and Communication Technology, ICEEICT 2021. https://doi.org/10.1109/ICEEICT53905.2021.9667810

Islam, M. M., Razzaque, M. A., & Almogren, A. (2020). A Secure Data Communication Framework for Underwater Acoustic Sensor Networks Using ECC-Based Authentication. *IEEE Access*, *8*, 78187–78201. https://doi.org/10.1109/ACCESS.2020.2989465

ISO/IEC/IEEE 29148-2018: Systems and software engineering — Life cycle processes — Requirements engineering (Número ISO/IEC/IEEE 29148-2018). (2018). https://doi.org/10.1109/IEEESTD.2018.8559686

Jensen, F. B., Kuperman, W. A., Porter, M. B., & Schmidt, H. (2011). *Computational ocean acoustics (2nd ed.)*. https://doi.org/10.1007/978-1-4419-8678-8

Kebapci, B., Mutlu, G., Baglica, I., Tosun, A., Ergin, S., Levent, V. E., Uysal, M., Paglierani, P., Pelekanakis, K., Petroccia, R., & Alves, J. (2022). Real-Time Implementation of an Underwater Quantum Key Distribution System. *2022 6th Underwater Communications and Networking Conference, UComms 2022*. https://doi.org/10.1109/UComms56954.2022.9905688

Khan, M. K., & Alghathbar, K. (2015). Security issues and challenges in underwater wireless sensor networks. *International Journal of Distributed Sensor Networks*, *2015*, 1–11. https://doi.org/10.1155/2015/123482

Khan, M., & Kim, D. (2022). Lightweight security mechanisms for underwater wireless sensor networks. *IEEE Internet of Things Journal*, *9*(6), 4231–4243. https://doi.org/10.1109/JIOT.2021.3112345

Krivokapić, D., & Vukotic, I. (2023). Security and privacy challenges in underwater sensor networks. *IEEE Access*, *11*, 23411–23430. https://doi.org/10.1109/ACCESS.2023.3259812

Lee, J., & Kim, D. (2010). A MAC protocol for underwater acoustic sensor networks. *IEEE Sensors Journal*, *10*(11), 1769–1776. https://doi.org/10.1109/JSEN.2010.2046742

Liu, L., Zhou, S., & Cui, J.-H. (2008). Prospects and problems of wireless communication for underwater sensor networks. *Wireless Communications and Mobile Computing*, *8*(8), 977–994. https://doi.org/10.1002/wcm.654

Lloret Mauri, J., IEEE Communications Society, IEEE Systems, M., Annual IEEE Computer Conference, International Conference on Advances in Computing, C. and I. (ICACCI) 4 2015. 08. 10-13 K., International Symposium on Emerging Topics in Circuits and Systems (SET-CAS) 2015.08.10 Kochi, K., International Symposium on Control, A., International Symposium on Recent Advances in Medical Informatics (RAMI) 4 2015.08.10 Kochi, K., & International Symposium on Natural Language Processing (NLP) 4 2015.08.13 Kochi, K. (2015). *International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2015 10-13 Aug. 2015, SCMS, Aluva, Kochi, Kerala, India ; [including co-affiliated symposia]*.

Louza, F. B., & Jesus, S. M. (2022). Low probability of detection underwater communications using a vector sensor. *2022 6th Underwater Communications and Networking Conference, UComms 2022*. https://doi.org/10.1109/UComms56954.2022.9905685

Luo, Z., Ding, Z., & Wang, S. (2023a). *Security authentication protocol for underwater sensor networks based on NTRU and position awareness*. https://doi.org/10.3390/jmse11040742

Luo, Z., Ding, Z., & Wang, S. (2023b). *Security authentication protocol for underwater sensor networks based on NTRU and position awareness*. https://doi.org/10.3390/jmse11040742

Marine Technology Society, Oceanic Engineering Society, Annual IEEE Computer Conference, & Oceans MTS/IEEE Conference 2012.05.21-24 Yeosu. (2012). *Oceans, 2012 - Yeosu 21-24 May 2012, the Ocean Resort, Yeosu, Republic of Korea ; conference*.

Mertens, J. S., Panebianco, A., Surudhi, A., Prabagarane, N., & Galluccio, L. (2023). Network intelligence vs. jamming in underwater networks: how learning can cope with misbehavior. *Frontiers in Communications and Networks*, *4*, 1179626. https://doi.org/10.3389/frcmn.2023.1179626

National Institute of Standards, & Technology. (2001). *Advanced Encryption Standard (AES)* (Número FIPS 197). https://doi.org/10.6028/NIST.FIPS.197

National Institute of Standards, & Technology. (2023). *Advanced Encryption Standard (AES)* (Número FIPS 197 (Update 1)). https://doi.org/10.6028/NIST.FIPS.197-upd1

NIST. (2023a). *FIPS 186-5: Digital Signature Standard (DSS)*. https://doi.org/10.6028/NIST.FIPS.186-5

NIST. (2023b). *FIPS 186-5: Digital Signature Standard (DSS)*. https://doi.org/10.6028/NIST.FIPS.186-5

Pal, S., Misra, S., & Das, S. K. (2022). Energy-efficient routing techniques for underwater wireless sensor networks. *IEEE Communications Surveys & Tutorials*, *24*(1), 147–176. https://doi.org/10.1109/COMST.2021.3123415

Rahman, M., & Kim, K. (2022). *A hybrid model for intrusion detection in IoT applications*. https://doi.org/10.1155/2022/4553502

Ranjan, R., Das, S., & Misra, S. (2022). *A lightweight secure scheme for underwater wireless acoustic networks*. https://doi.org/10.3390/jmse10050831

Saeed, N., Celik, A., & Al-Naffouri, T. Y. (2023). Underwater wireless sensor networks: A sustainability perspective. *Sustainability*, *15*(9), 7198. https://doi.org/10.3390/su15097198

Securosys. (s/f). *Elliptic Curve Integrated Encryption Scheme (ECIES)*. Recuperado https://docs.securosys.com/pkcs/category/ecies/

Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal*, *27*(3), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Shin, S. Y., & Park, S. H. (2008). Omnet++ based simulation for underwater environment. *Proceedings of The 5th International Conference on Embedded and Ubiquitous Computing, EUC 2008*, *2*, 689–694. https://doi.org/10.1109/EUC.2008.34

Signori, F., & Casari, P. (2022). Recent advances in underwater wireless sensor networks. *IEEE Access*, *10*, 11521–11544. https://doi.org/10.1109/ACCESS.2022.3141234

Souza, R., & Vieira, L. (2013). Data aggregation techniques for underwater sensor networks. *Sensors*, *13*(5), 6623–6656. https://doi.org/10.3390/s130506623

Standards for Efficient Cryptography Group (SECG). (2009). *SEC 1: Elliptic Curve Cryptography* (Número Version 2.0). https://www.secg.org/sec1-v2.pdf

Stojanovic, M., & Chitre, M. (2018). *Underwater acoustic communication*.

Technische Universität Wien, Institute of Electrical and Electronics Engineers, IEEE Communications Society, & Han'guk T'ongsin Hakhoe. (2016). *ICUFN 2016 : the Eighth International Conference on Ubiquitous and Future Networks : July 5(Tue.)-July 8(Fri.), 2016, Technische Universität (TU) Wien, Vienna, Austria.*

Urick, R. J. (1983). *Principles of Underwater Sound* (3a ed.). McGraw-Hill.

Velázquez, R., Gutiérrez, J., & Higuera-Toledano, M. (2021). Underwater Acoustic Communication Security Using Symmetric and Asymmetric Cryptography. *IEEE Latin America Transactions*, *19*(5), 908–915. https://doi.org/10.1109/TLA.2021.9474691

Wahid, A., Kim, D., & Lee, H. J. (2021). *A lightweight authenticated encryption scheme for underwater sensor networks*. https://doi.org/10.3390/s21020425

Xie, C., Liu, Y., & Liu, X. (2021a). Design and Implementation of Underwater Wireless Sensor Network Simulation Platform Based on OMNeT++. *IEEE Access*, *9*, 138714–138724. https://doi.org/10.1109/ACCESS.2021.3118843

Xie, C., Liu, Y., & Liu, X. (2021b). Design and Implementation of Underwater Wireless Sensor Network Simulation Platform Based on OMNeT++. *IEEE Access*, *9*, 138714–138724. https://doi.org/10.1109/ACCESS.2021.3118843

Yu, D., & Lee, J. (2023). Challenges in Secure Underwater Wireless Sensor Network with Fine-grained Access Control. *International Conference on ICT Convergence*, 900–902. https://doi.org/10.1109/ICTC58733.2023.10393842

Yun, S., & Park, S. (2016). Energy-efficient communication in underwater sensor networks. *IEEE Sensors Journal*, *16*(11), 4077–4084. https://doi.org/10.1109/JSEN.2016.2535312

Zhang, Q., Zhang, P., & Wu, J. (2022). *Cryptography-based secure underwater acoustic communication: A survey*. https://doi.org/10.3390/electronics11122415

Alharbi, A. A., & Ibrahim, K. M. (2026). Enhancing UWSN security with an improved depth-based routing protocol against depth spoofing attacks. Sensors, 26(1), 17. https://doi.org/10.3390/s26010017

Mertens, J. S., Panebianco, A., Surudhi, A., Prabagarane, N., & Galluccio, L. (2023). Network intelligence vs. jamming in underwater networks: How learning can cope with misbehavior. Frontiers in Communications and Networks, 4, 1179626. https://doi.org/10.3389/frcmn.2023.1179626

European Commission. (2024). Commission Recommendation (EU) 2024/779 of 26 February 2024 on secure and resilient submarine cable infrastructures. Official Journal of the European Union. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L_202400779

European Commission & High Representative of the Union for Foreign Affairs and Security Policy. (2025). Joint Communication: EU action plan on cable security. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52025JC0009

Associated Press. (2025, January 30). Sweden rules out sabotage in Baltic undersea data cable rupture and releases a ship. Associated Press.

Anexos

Anexo A. Ficha de requerimientos

ANEXO A. Ficha de requerimientos

Proyecto de titulación: Adaptación de un esquema híbrido basado en ECDH + KDF (SHA-256) + AES-CTR + firma ECDSA para redes UWSN en OMNeT++

Objetivo del anexo: Consolidar y presentar, en un formato unificado, los requerimientos definidos en la Sección 3.2 (StSR, SySR y SRSH), incluyendo trazabilidad y guía de verificación/validación.

Fecha de realización: Enero 2026

Descripción y abreviaturas

Descripción	Abreviatura
Requerimientos de Stakeholders	StSR
Requerimientos del Sistema	SySR
Requerimientos de Hardware y Software	SRSH

Requerimientos de Stakeholders (StSR)

Nota: los requerimientos de stakeholders consolidan necesidades funcionales del sistema UWSN y criterios de evaluación descritos en la literatura de UWSN y seguridad (Chitre et al., 2008; Stojanovic & Chitre, 2018; Zhang et al., 2022).

Nomenclatura	Requerimiento	Descripción	Prioridad
StSR-01	Permitir proteger datos transmitidos en una red UWSN	El proyecto deberá permitir proteger datos transmitidos en una red UWSN mediante un esquema criptográfico basado en ECC/ECIES.	Alta
StSR-02	Permitir evaluar el impacto del cifrado	El proyecto deberá permitir evaluar el impacto del cifrado en el desempeño de la red (latencia, entrega, velocidad de transmisión y procesamiento).	Alta
StSR-03	Integrarse como un módulo funcional	El sistema deberá integrarse como un módulo funcional (Escenario de	Alta

		simulación) dentro de un entorno de simulación de redes (OMNeT++).	
StSR-04	Permitir comparación entre un escenario base	El proyecto deberá permitir comparación entre un escenario base (sin cifrado) y uno seguro (con ECIES).	Alta
StSR-05	Contemplar una arquitectura con nodos sensores y una boya/sink/gateway	El proyecto deberá contemplar una arquitectura con nodos sensores y una boya/sink como receptor central.	Alta
StSR-06	Permitir escalabilidad del número de nodos para analizar su comportamiento	El proyecto deberá permitir escalabilidad del número de nodos para analizar	Media

		comportamiento bajo diferentes densidades.	
StSR-07	Generar resultados reproducibles	El proyecto deberá generar resultados reproducibles (parámetros, escenarios y semillas documentadas).	Alta
StSR-08	La solución propuesta deberá privilegiar eficiencia en entornos con recursos limitados	La solución propuesta deberá privilegiar eficiencia en entornos con recursos limitados, justificando el uso de ECC por tamaño de claves y costo computacional.	Alta
StSR-09	La implementación deberá ser modular y	La implementación deberá ser modular y extensible para permitir futuras	Media

	extensible para permitir futuras mejoras.	variantes (curvas alternativas, ataques simulados, movilidad).	
StSR-10	El desarrollo deberá mantener claridad metodológica	El desarrollo deberá mantener claridad metodológica (toma de requerimientos, trazabilidad y criterios de validación).	Alta
StSR-11	Representar un canal subacuático con condiciones que afecten la comunicación	El sistema deberá representar un canal subacuático con condiciones que afecten la comunicación (atenuación, retardo, ruido) para analizar el impacto real del cifrado.	Alta

Restricciones y supuestos (StSR)

Nomenclatura	Requerimiento	Descripción	Prioridad
StSR-12	La validación se realizará mediante simulación y no mediante hardware	La validación se realizará mediante simulación y no mediante hardware real (alcance del trabajo).	Alta
StSR-13	Se asume un esquema con receptor central	Se asume un esquema con receptor central (boya/sink) y nodos sensores que envían datos hacia él (topología base).	Alta
StSR-14	Se asume que las claves públicas necesarias para ECIES están.	Se asume que las claves públicas necesarias para ECIES están disponibles para los nodos según el escenario (preconfiguración en simulación).	Media

StSR-15	Se prioriza un esquema híbrido	Se prioriza un esquema híbrido (ECC) confidencialidad con AES-CTR + integridad/autenticación con firma ECDSA (Velázquez et al., 2021; Zhang et al., 2022) .	Alta

Requerimientos del sistema - funcionales (SySR)

Nomenclatura	Requerimiento	Descripción	Prioridad
SySR-01	Implementar ECIES para cifrado híbrido	El sistema deberá implementar ECIES para cifrado híbrido, incluyendo componente ECC y un esquema simétrico para confidencialidad (AES-CTR) e integridad/autenticación mediante firma digital ECDSA (SHA-256).	Alta

SySR-02	Generar una clave efímera por sesión o por mensaje	El sistema deberá generar una clave efímera por sesión o por mensaje, y calcular el secreto compartido mediante ECIES.	Alta
SySR-03	Derivar claves simétricas a partir del secreto compartido usando una...	El sistema deberá derivar claves simétricas a partir del secreto compartido usando una función hash/KDF (p. ej., SHA-256).	Alta
SySR-04	Cifrar la carga útil usando AES en modo CTR y...	El sistema deberá cifrar la carga útil usando AES en modo CTR y generar una firma digital ECDSA (SHA-256) sobre el ciphertext y metadatos relevantes (por ejemplo, IV/nonce y	Alta

		contador/identificador anti-replay cuando aplique).	
SySR-05	Encapsular y transmitir un mensaje seguro con al menos: R	El sistema deberá encapsular y transmitir un mensaje seguro con al menos: R (punto público efímero), IV/nonce/contador, C (ciphertext) y Sig (firma ECDSA).	Alta
SySR-06	Reconstruir el secreto	El receptor (boya/sink) deberá reconstruir el secreto, verificar la firma ECDSA y descifrar; si la verificación falla, deberá descartar el paquete.	Alta
SySR-07	El módulo criptográfico deberá integrarse a la capa de aplicación...	El módulo criptográfico deberá integrarse a la capa de aplicación del modelo OMNeT++	Alta

		para cifrar antes de transmitir y descifrar al recibir.	
SySR-08	Registrar métricas por evento: tiempo de cifrado/descifrado	El sistema deberá registrar métricas por evento: tiempo de cifrado/descifrado, éxito de verificación y latencia total por paquete (Batool & Ahmad, 2021).	Alta
SySR-09	Permitir habilitar/deshabilitar el módulo de seguridad para obtener escenarios sin...	El sistema deberá permitir habilitar/deshabilitar el módulo de seguridad para obtener escenarios sin seguridad vs seguro.	Media
SySR-10	Soportar múltiples tamaños de red	El sistema deberá soportar múltiples tamaños de red (escenarios escalables) definidos en archivos de configuración.	Media

Requerimientos del sistema - no funcionales (SySR)

Nomenclatura	Requerimiento	Descripción	Prioridad
SySR-11	Mantener una sobrecarga de seguridad medible y reportable	El sistema deberá mantener una sobrecarga de seguridad medible y reportable, para evaluar viabilidad en UWSN (Akyildiz et al., 2005b; Barbeau & Garcia-Alfaro, 2015).	Alta
SySR-12	El módulo criptográfico deberá ser modular	El módulo criptográfico deberá ser modular (bajo acoplamiento), permitiendo sustitución de curva o modo de cifrado sin rediseñar toda la red (Casari & Zorzi, 2022; Domingo, 2011).	Media

SySR-13	Ser robusto ante pérdida/colisiones: fallos de recepción o verificación no...	El sistema deberá ser robusto ante pérdida/colisiones: fallos de recepción o verificación no deben detener la simulación.	Alta
SySR-14	El modelo deberá permitir reproducibilidad experimental	El modelo deberá permitir reproducibilidad experimental (parámetros documentados, escenarios repetibles).	Alta

Requerimientos del sistema - interfaces y comunicación (SySR)

Nomenclatura	Requerimiento	Descripción	Prioridad
SySR-15	Definir el formato del mensaje seguro y su serialización	El sistema deberá definir el formato del mensaje seguro y su serialización (R, C, Sig y metadatos como IV/nonce y nonce/contador/seq si aplica).	Alta

SySR-16	Interoperar con el stack de red del entorno	El módulo deberá interoperar con el stack de red del entorno (INET/UAN) sin romper el flujo normal de envío/recepción.	Alta
SySR-17	Permitir configurar por parámetros: longitud de claves	El sistema deberá permitir configurar por parámetros: longitud de claves, tamaño de payload, tasa de envío y número de nodos.	Media
SySR-18	Disponer de un mecanismo para acceder a su clave privada...	El receptor deberá disponer de un mecanismo para acceder a su clave privada y a la información necesaria para descifrar de forma consistente.	Alta

Requerimientos del sistema - seguridad (SySR)

Nomenclatura	Requerimiento	Descripción	Prioridad
SySR-19	Garantizar confidencialidad del payload transmitido.	El sistema deberá garantizar confidencialidad del payload transmitido.	Alta
SySR-20	Garantizar integridad y autenticación de mensajes mediante etiqueta criptográfica verificable.	El sistema deberá garantizar integridad y autenticación de mensajes mediante etiqueta criptográfica verificable.	Alta
SySR-21	Mitigar replay mediante nonces/contadores o mecanismo equivalente	El sistema deberá mitigar replay mediante nonces/contadores o mecanismo equivalente (en simulación).	Media
SySR-22	El esquema deberá proveer secreto hacia adelante	El esquema deberá proveer secreto hacia adelante	Media

		(forward secrecy) usando componentes efímeros en ECIES.	
SySR-23	Permitir modelar o discutir resistencia frente a eavesdropping y manipulación...	El sistema deberá permitir modelar o discutir resistencia frente a eavesdropping y manipulación en el enlace acústico (amenazas base).	Alta

Requerimientos de software (SRSH)

Nomenclatura	Requerimiento	Descripción	Prioridad
SRSH-01	Disponer de un entorno de simulación OMNeT++ para modelado de...	Se deberá disponer de un entorno de simulación OMNeT++ para modelado de red.	Alta
SRSH-02	Contar con INET Framework para soporte de capas y modelos...	Se deberá contar con INET Framework para soporte de capas y modelos de red.	Alta

SRSH-03	Disponer de un modelo/extensión tipo UAN	Se deberá disponer de un modelo/extensión tipo UAN (Underwater Acoustic Network) para enlaces acústicos.	Alta
SRSH-04	Disponer de un entorno C++	Se deberá disponer de un entorno C++ (compilador y depurador) para implementar el módulo criptográfico.	Alta
SRSH-05	Disponer de herramientas para exportar y procesar métricas	Se deberá disponer de herramientas para exportar y procesar métricas (IDE OMNeT++ / CSV / hojas de cálculo).	Media

Requerimientos del entorno de simulación (SRSH)

Nomenclatura	Requerimiento	Descripción	Prioridad

SRSH-06	El entorno deberá permitir configurar parámetros del canal acústico: retardo...	El entorno deberá permitir configurar parámetros del canal acústico: retardo de propagación, atenuación y ruido, para análisis realista (Barbeau & Garcia-Alfaro, 2015).	Alta
SRSH-07	El entorno deberá permitir ejecución de múltiples escenarios	El entorno deberá permitir ejecución de múltiples escenarios (variación de nodos/tráfico) y recolección de métricas por corrida.	Alta
SRSH-08	El entorno deberá permitir integrar módulos personalizados a nivel de...	El entorno deberá permitir integrar módulos personalizados a nivel de aplicación para instrumentación de métricas criptográficas.	Alta

Requerimientos de hardware/plataforma de ejecución (SRSH)

Nomenclatura	Requerimiento	Descripción	Prioridad
SRSH-09	El equipo de ejecución deberá contar con CPU multi-núcleo	El equipo de ejecución deberá contar con CPU multi-núcleo (≥ 4 núcleos) para ejecutar escenarios con decenas de nodos en tiempo razonable.	Media
SRSH-10	El equipo deberá contar con memoria RAM suficiente	El equipo deberá contar con memoria RAM suficiente (≥ 8 GB) para compilar y simular escenarios UWSN con INET/UAN.	Media
SRSH-11	Contar con almacenamiento suficiente para resultados y trazas	Se deberá contar con almacenamiento suficiente para	Media

	resultados y trazas (archivos de salida y vectores).	

Matriz de trazabilidad (extracto)

StSR	SySR relacionados	SRSH relacionados
StSR-01	SySR-01, SySR-02, SySR-03, SySR-04, SySR-05, SySR-06	SRSH-01, SRSH-04
StSR-02	SySR-08, SySR-10, SySR-11	SRSH-05, SRSH-07
StSR-03	SySR-07, SySR-16	SRSH-01, SRSH-02
StSR-04	SySR-09, SySR-10	SRSH-07
StSR-11	SySR-13, SySR-16	SRSH-03, SRSH-06

Métodos de verificación y validación por tipo de requerimiento (guía)

Tipo de requerimiento	Método principal	Evidencia esperada
StSR	ID + AN	Matriz de trazabilidad, justificación teórica

SySR funcional	PS + DM	Escenarios controlados y logs de cifrado/descifrado
SySR seguridad	PS + AN	Pruebas de integridad (firma ECDSA), replay si aplica, análisis de amenazas
SRSH	ID + DM	Versiones, instalación, configuración y reproducibilidad

Elaborado por: Andrés Vasconez

Anexo B. Instalación de Lubuntu

Iniciando el instalador

Al iniciar el proceso de arranque, se mostrará una pantalla de inicio de GRUB. Para instalar, seleccione " Probar" o "Instalar Lubuntu" . Si tiene una tarjeta gráfica con problemas, seleccione "Lubuntu (gráficos seguros)" (por ejemplo, algunas tarjetas NVIDIA lo requieren). Para probar la RAM, seleccione "Probar memoria" .

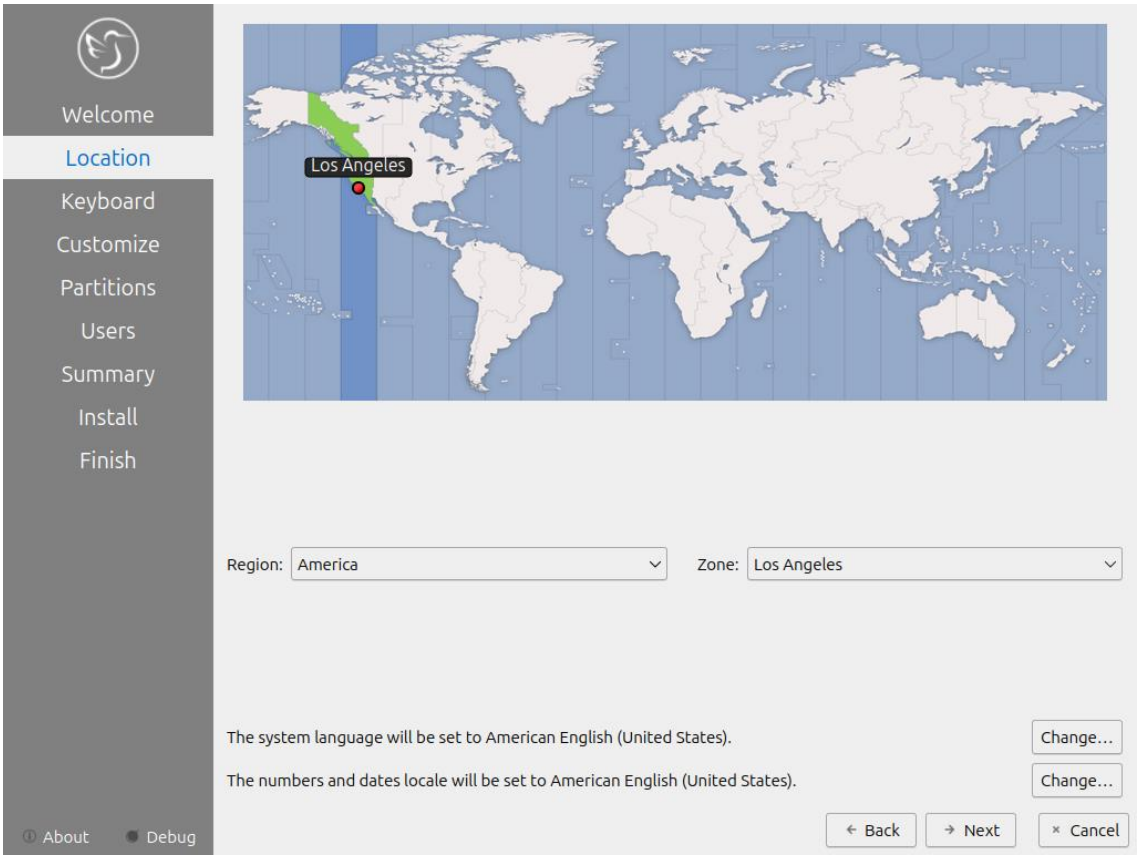
La siguiente pantalla le presentará una opción entre Probar Lubuntu , que le dará una sesión en vivo para ejecutar Lubuntu como mejor le parezca, o ejecutar el instalador directamente con Instalar Lubuntu . Sobre esto hay un campo para seleccionar el idioma que desea instalar en el menú desplegable Seleccionar su idioma . Para obtener esta pantalla en su idioma, seleccione uno y luego presione el botón Confirmar . Si necesita configuraciones de red más avanzadas para instalar Lubuntu que simplemente conectarse a Wi-Fi, use la opción Probar Lubuntu . Si elige Instalar Lubuntu, el instalador se iniciará de inmediato y se instalará en su idioma. En la parte inferior de la pantalla, si está conectado a Internet automáticamente como con un cable Ethernet, dirá Conectado en la parte inferior. Si no está conectado a Internet en la parte inferior dirá No conectado .

Una vez iniciada la sesión en vivo, explore Lubuntu y asegúrese de que todo su hardware funciona correctamente. Una vez listo para instalar Lubuntu, haga doble clic en el icono en la esquina superior izquierda del escritorio: Instalar Lubuntu 25.04 o Herramientas del sistema ▸ Instalar Lubuntu 25.04 .

Accederá a la pantalla de bienvenida del instalador de Lubuntu. Puede cambiar el idioma del instalador en el menú desplegable "Idioma" . Tras seleccionar el idioma, el botón "Siguiente" le llevará a la siguiente tarea. Para cancelar una instalación, pulse el botón "Cancelar" . A la izquierda del instalador encontrará una barra lateral que muestra el paso de la instalación en el que se encuentra.

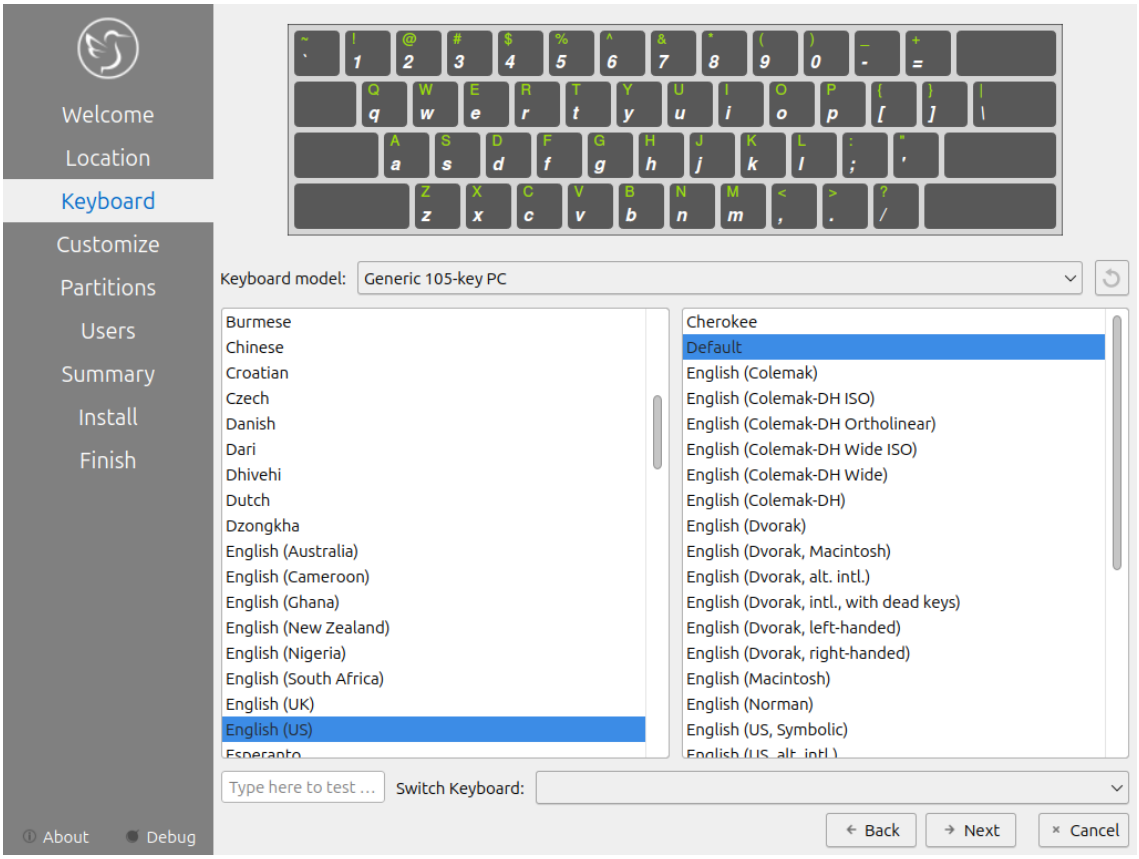
Seleccionar su ubicación

La siguiente pantalla te mostrará un mapa del mundo donde podrás elegir tu ubicación. Esta se utilizará para configurar tu zona horaria y el servidor de descarga. Para ver la región que has elegido, utiliza el menú desplegable " Región" . El campo "Zona" debe tener una ciudad importante con la misma hora que tú. Si quieres acceder a tu zona horaria manualmente más rápido, puedes escribir el nombre de una ciudad importante en la misma zona horaria. En la parte inferior, puedes cambiar el idioma de tu sistema pulsando el botón "Cambiar" de la parte superior . Para cambiar la apariencia de los números y las fechas en tu sistema, pulsa el botón "Cambiar" de la parte inferior . Para cambiar ciertas teclas del teclado, utiliza el menú desplegable "Cambiar teclado" . Para ir al siguiente paso, pulsa de nuevo el botón "Siguiente" . Para volver a seleccionar tu idioma y cambiarlo, pulsa el botón "Atrás" .



Seleccionar la distribución del teclado

A continuación, seleccione la distribución de su teclado para comprobar que coincide con la imagen. El menú "Modelo de teclado" le permite elegir diferentes variantes, la columna izquierda le permite cambiar el idioma y la columna derecha le permite elegir diferentes variantes. En la parte inferior, puede escribir para comprobar que la distribución es correcta. Para comprobar si el teclado funciona correctamente, escriba "Escribir aquí" . Una vez seleccionada la distribución, pulse el botón "Siguiente" . Para obtener una vista previa de la distribución, mire la parte superior de la ventana.



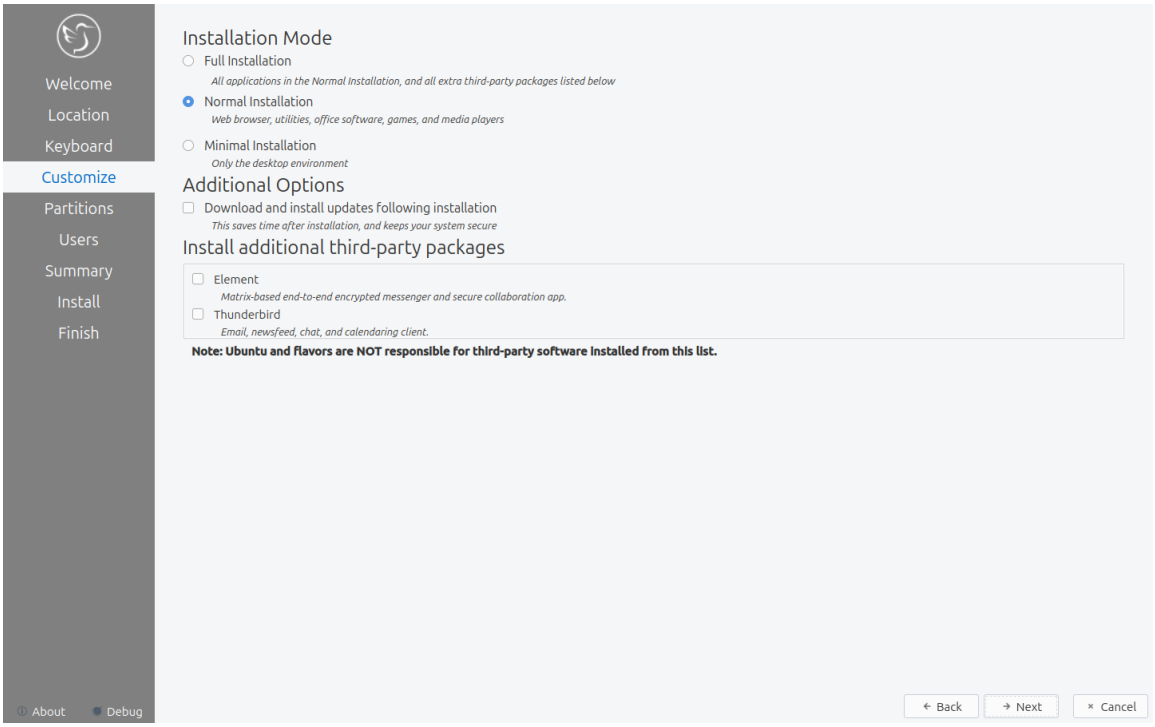
Personalizar

A continuación, podrá elegir qué aplicaciones instalar, controladores de terceros adicionales o instalar actualizaciones durante la instalación. Para instalar todas las aplicaciones y la lista de aplicaciones de terceros, seleccione " Instalación completa" . Para seleccionar las aplicaciones normales incluidas en Lubuntu, seleccione el botón " Instalación normal" . Para elegir una instalación mínima que solo incluya el entorno de escritorio, seleccione el botón "Instalación mínima" . Para instalar actualizaciones durante la instalación, marque la casilla "Descargar e instalar actualizaciones después de la instalación ". Para instalar controladores de terceros, e incluso propietarios, marque la casilla "Instalar software de terceros para gráficos, hardware WiFi y formatos multimedia adicionales" .

Seleccione las aplicaciones adicionales que desee instalar. Hay casillas de verificación para cada programa en " Instalar paquetes adicionales de terceros" . Para instalar Element

para el chat cifrado de Matrix, marque la casilla Element . Para instalar el cliente de correo electrónico y calendario Thunderbird, marque la casilla Thunderbird .

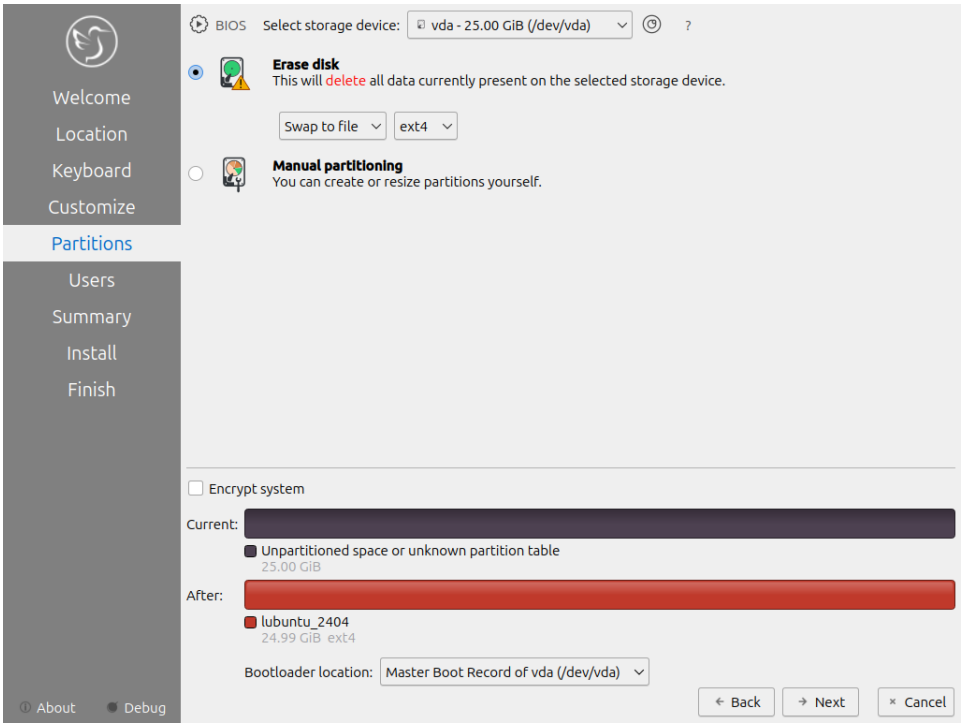
Para ir a la siguiente parte de la instalación, pulse el botón Siguiente . Para volver a seleccionar el teclado, pulse el botón Atrás .



Configuración de particiones

Si solo desea tener Lubuntu en su equipo, puede seleccionar el botón Borrar disco . Esto formateará el disco y eliminará todos los datos que contiene , por lo que es fundamental realizar una copia de seguridad de sus datos antes de este punto. Si elige hacer esto, puede continuar con la configuración del usuario. Para cambiar el dispositivo de almacenamiento en el que instalar Lubuntu, utilice el menú desplegable Seleccionar dispositivo de almacenamiento . Para ver si está instalando en modo UEFI o BIOS, se muestra en la esquina superior izquierda de esta ventana. Para elegir usar un archivo de intercambio en Borrar disco, mantenga Intercambio en archivo o, para no usar intercambio, elija Sin intercambio . Para elegir el sistema de archivos que desea usar, despliega el menú desplegable de la derecha para elegir el sistema de archivos que desea usar. Si desea cifrar su unidad, presione

la casilla de verificación Cifrar sistema y luego deberá ingresar la contraseña de cifrado dos veces para asegurarse de haberla escrito correctamente. Se recomienda encarecidamente anotar esta contraseña y guardarla en un lugar seguro.



Nota

Si tenía una instalación previa de Linux con una partición de intercambio, deberá desmontarla. Para ello, ejecute

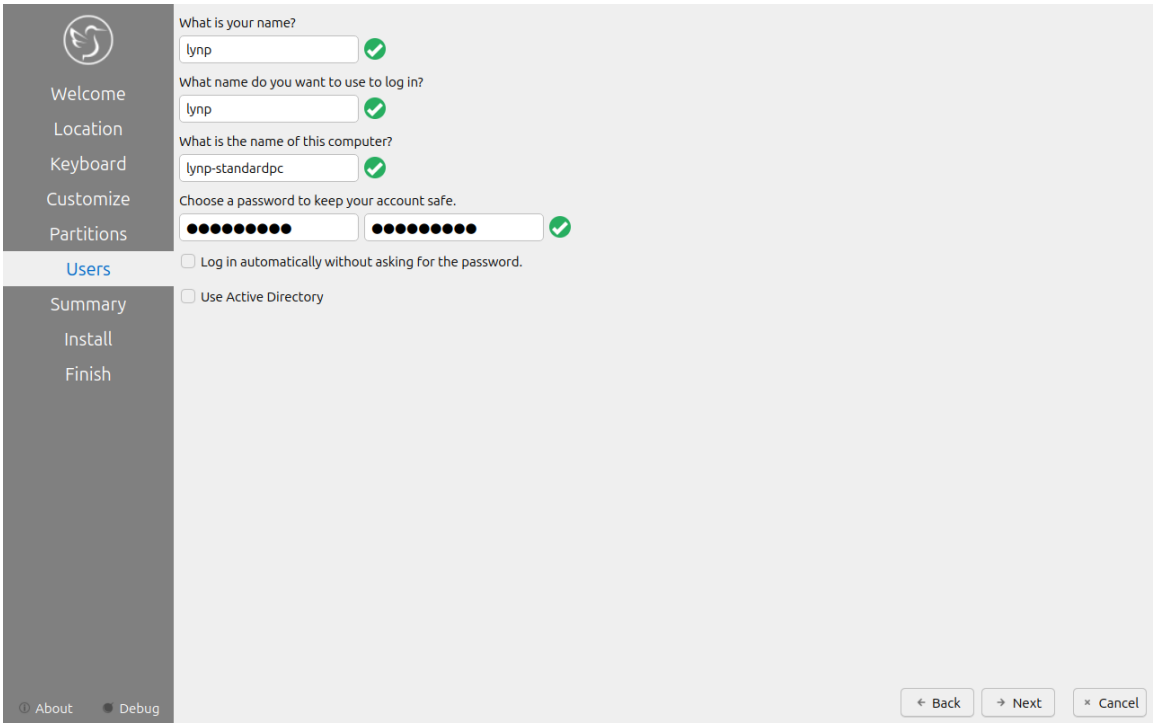
sudo swapoff -a; sudo swapon /dev/zram0

Esto las desmontará, junto con cualquier partición de intercambio, manteniendo zram habilitado. Esto no funcionará si tiene una partición de datos montada; abra PCManFM-Qt y presione la flecha hacia arriba en cada partición en la barra lateral de "Lugares" para desmontarlas todas.

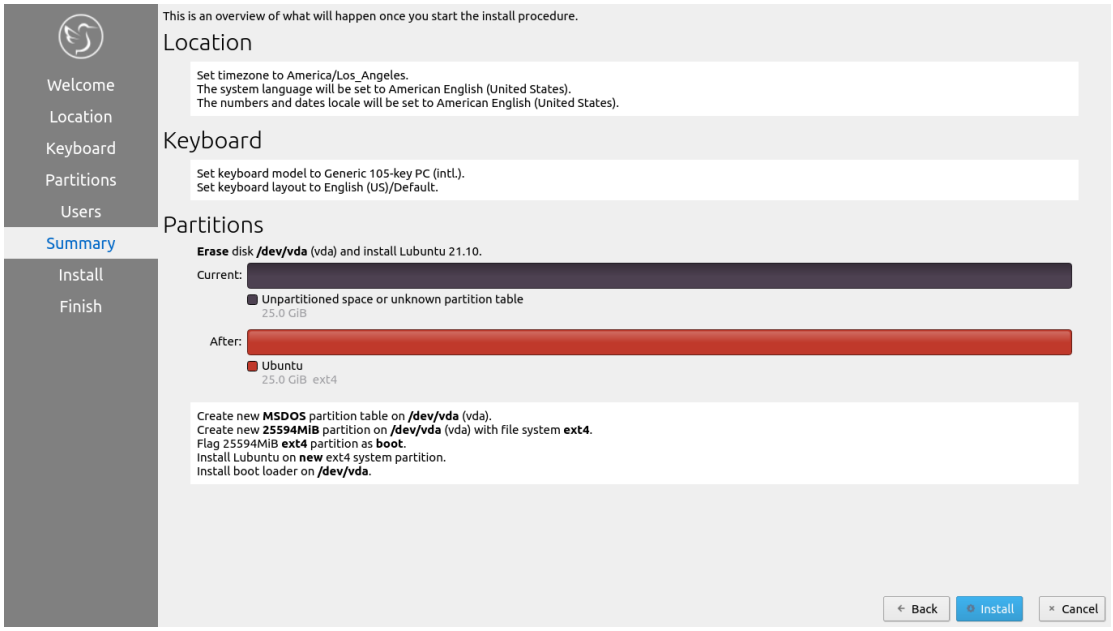
Para volver a la sección "Seleccionar la distribución del teclado", pulse el botón "Atrás" . Para avanzar a la sección "Configurar usuarios", pulse el botón "Siguiente" .

Configuración de usuario

La sección de configuración de usuario crea un perfil de usuario, que consiste en escribir su nombre en el campo ¿Cuál es su nombre?. A continuación, escriba su nombre de usuario en el campo ¿Qué nombre desea usar para iniciar sesión?. Coloque el nombre de host que desea para su computadora en el campo ¿Cuál es el nombre de esta computadora?. Lo último que debe ingresar es su contraseña en Elija una contraseña para mantener su cuenta segura . Ingrese su contraseña dos veces para asegurarse de no haberla escrito mal. A la derecha de su contraseña tendrá una x si sus contraseñas no coinciden. Si tiene una contraseña débil a la derecha tendrá un triángulo naranja que indica por qué su contraseña es débil. Las contraseñas de menos de 8 caracteres, que contienen una secuencia como 1234 o que se basan en una palabra simple del diccionario como testing1 se consideran débiles. Si un nombre realmente se puede usar como nombre de usuario o nombre de host, aparecerá una marca de verificación verde a su derecha; si no puede, aparecerá una x roja. Al presionar el botón Siguiente , obtendrá una pantalla de resumen, que le mostrará la configuración antes de que comience la instalación. Una vez que haya revisado el resumen, haga clic en el botón Instalar para comenzar la instalación.

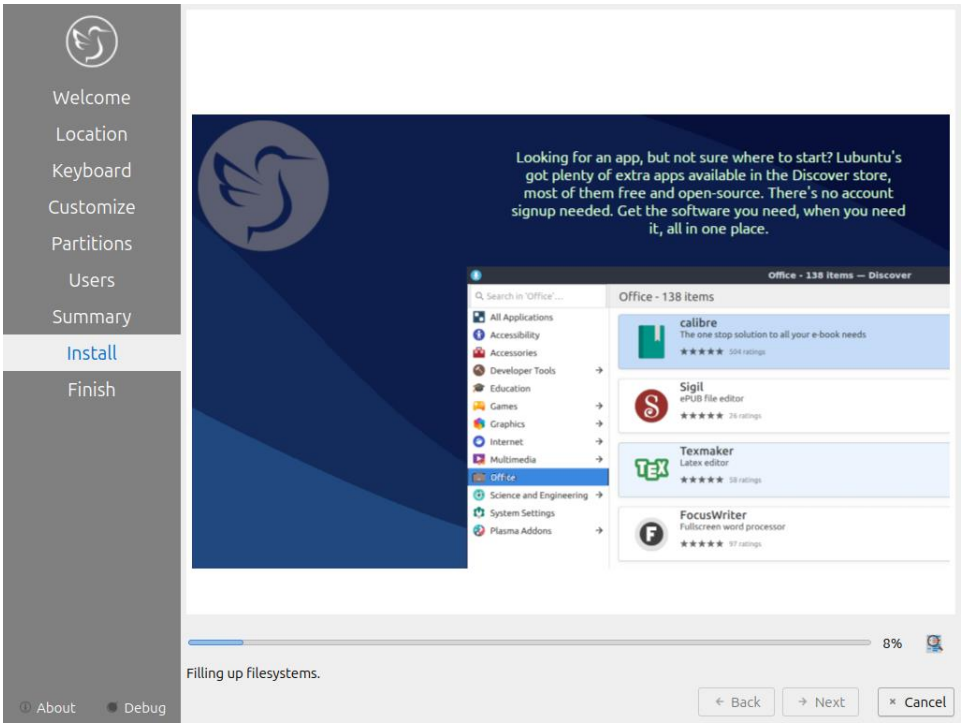


La pantalla de resumen muestra la configuración que se instalará y puede comprobar que todo esté como desea. Tras pulsar el botón Instalar , aparecerá un cuadro de diálogo para confirmar la instalación. Para instalar, pulse el botón Instalar ahora . Para no iniciar la instalación y volver atrás, pulse el botón Volver .



La instalación

El instalador de Lubuntu proporciona información útil durante su ejecución. En la parte inferior de la ventana hay una barra de progreso. A la derecha de esta barra se muestra el porcentaje de finalización de la instalación. Para ver el resultado de la instalación desde la línea de comandos, pulse el botón de la lupa . Una vez instalado Lubuntu, aparecerá una casilla de verificación " Reiniciar ahora" después de que Lubuntu se haya instalado y haya finalizado. Se muestra una presentación de diapositivas durante la instalación; para pasar a la siguiente diapositiva, haga clic con el botón izquierdo y para pasar a la anterior, haga clic con el botón derecho.

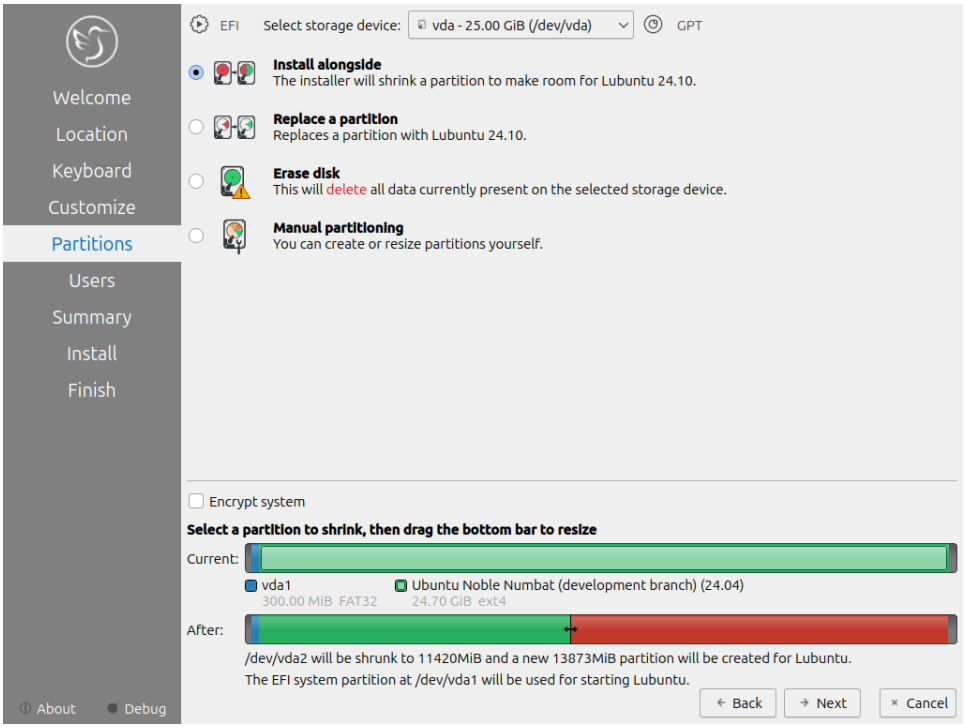


Reiniciando en la instalación finalizada

Una vez finalizada la instalación, deberá reiniciar el sistema con la nueva instalación. Si desea continuar usando el sistema en vivo, pero finalizar el instalador, desmarque la casilla " Reiniciar ahora" . Para cerrar el instalador, pulse el botón "Listo" . Después, el equipo se reiniciará y deberá retirar el medio de instalación.

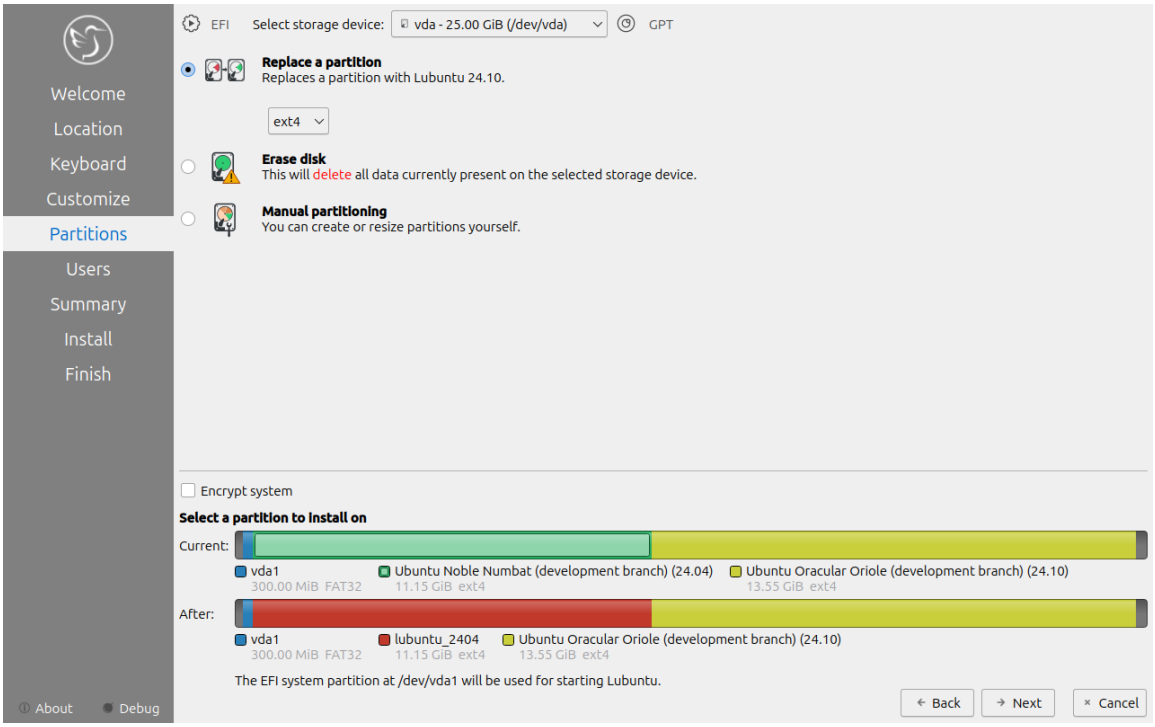
Instalar junto a

Para reducir el tamaño de la partición actual e instalar Lubuntu junto con una instalación existente y poder iniciar ambas, seleccione " Instalar junto" . El gráfico de barras "Actual" muestra el espacio que ocupa actualmente una partición. La sección "Después" muestra el espacio que ocuparán ambas instalaciones. Para cambiar el espacio que ocupará cada partición, deslice el centro hacia adelante y hacia atrás para distribuir el espacio entre ambas instalaciones. Si desea cifrar la instalación, marque la casilla "Cifrar sistema" .



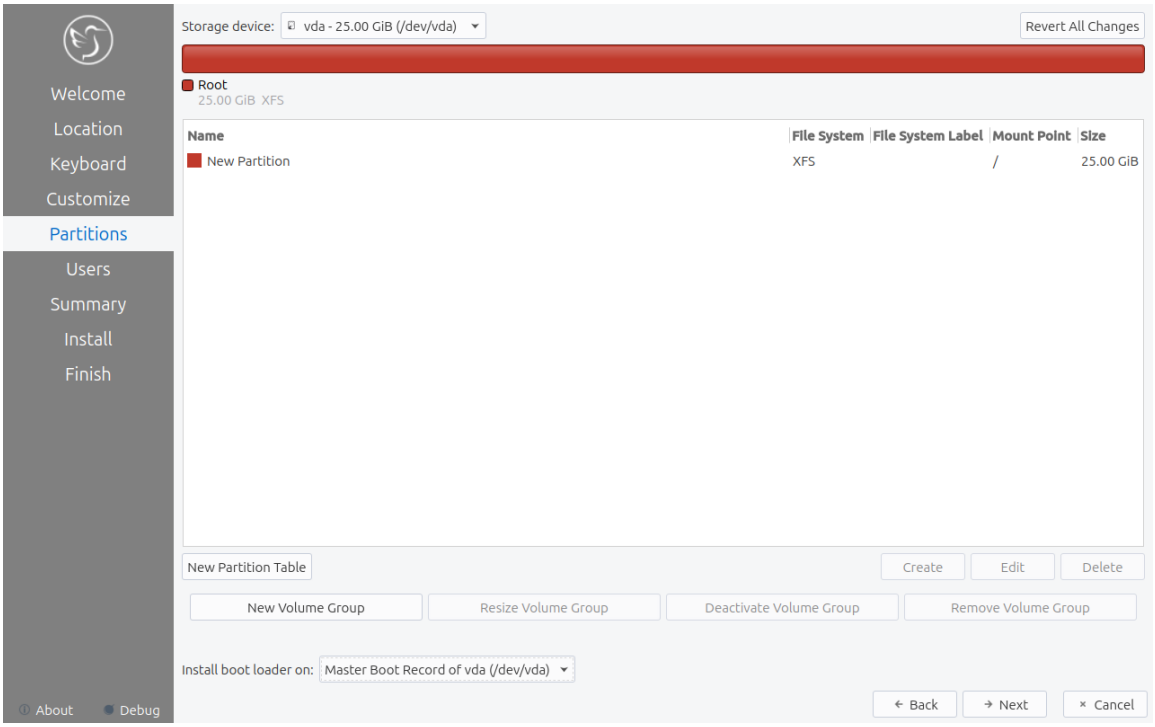
Reemplazar una partición

Para reemplazar una partición, seleccione Reemplazar una partición en la pestaña Particiones . Para seleccionar en qué sistema de archivos instalar Lubuntu, elija en el menú desplegable bajo Reemplazar una partición . Para seleccionar en qué unidad instalar, elija en el menú Seleccionar dispositivo de almacenamiento . Para cifrar su instalación de Lubuntu, marque la casilla de verificación Cifrar sistema . Para ver su diseño actual de particiones, mire el gráfico de barras Actual . Para ver qué partición aparece debajo del gráfico. Para seleccionar qué partición reemplazar, haga clic izquierdo en la partición en el gráfico Actual . Para ver qué particiones se utilizarán después de la instalación, lea el campo Después . Para ver el tamaño y el sistema de archivos de todas las particiones, vea el gráfico de barras Después . Para ver desde dónde arrancará Lubuntu también está en la parte inferior. Para continuar con la instalación, presione Siguiente .



Particionado manual

Si desea configurar particiones manualmente, como opción avanzada, deberá elegir el sistema de archivos que desee. Un sistema de archivos controla cómo se accede a sus archivos en los niveles inferiores del disco. Si inicia su computadora en modo UEFI, un firmware más moderno que BIOS, deberá crear una partición de sistema EFI (consulte la sección "Particiones de sistema EFI" en Wikipedia para obtener más información). Para crear esta partición, necesitará un sistema de archivos FAT32 con el indicador ESP montado en /boot/efi/, en el punto de montaje. También necesitará un sistema de archivos raíz (/). Lubuntu incluye varios sistemas de archivos: Ext4, XFS y Btrfs. Lubuntu 22.04 presenta un error al instalar BTRFS. Esta guía sobre Lubuntu...

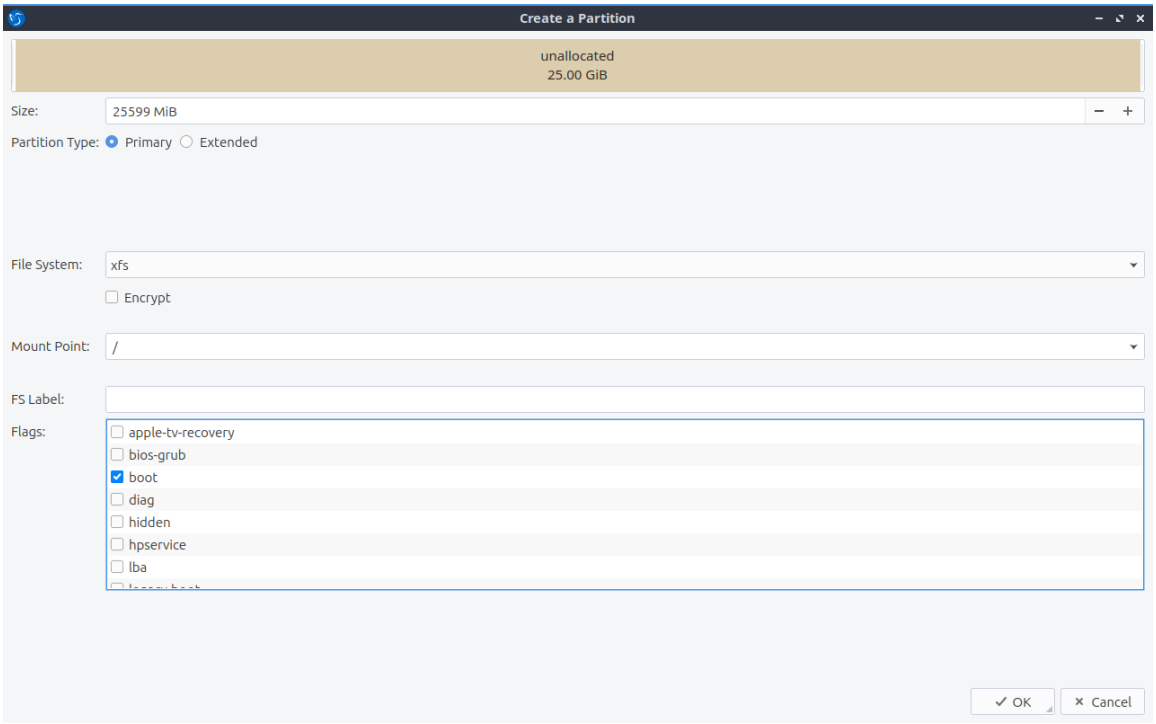


Si tiene un disco duro o una unidad de estado sólido nuevos, pulse el botón "Nueva tabla de particiones" . Esto eliminará todo el disco si contiene datos. Tras pulsarlo, aparecerá un cuadro de diálogo que indica el tipo de tabla de particiones que debe usar. El botón "Registro de arranque maestro" creará una tabla de particiones antigua, pero solo permitirá 4 particiones primarias y particiones de hasta 2 terabytes. El botón "Tabla de particiones GUID" funciona con discos grandes, pero es posible que no lo reconozcan los sistemas operativos antiguos. Para volver a la ventana principal de particiones, pulse el botón "Aceptar" .

Para cambiar el disco que está particionando, utilice el menú desplegable Dispositivo de almacenamiento .

Puede crear una partición haciendo clic en el botón Crear , lo que abrirá un cuadro de diálogo. El campo Sistema de archivos es un menú desplegable; seleccione el sistema de archivos que desee. También debe seleccionar dónde desea montar la partición en el menú desplegable Punto de montaje . Para cambiar el tamaño de la partición, cambie el campo Tamaño . Para cambiar el tamaño o el tipo de una partición después de crearla

inicialmente, presione el botón Editar . Para eliminar una partición, presione el botón Eliminar . Necesitará al menos una partición raíz (/) y, si está arrancando un sistema EFI, también necesitará una partición montada en /boot/efi. Otra opción común es tener todos sus datos en su propia partición, que puede incluso estar en su propio disco físico separado, que puede montarse en /home. Si desea cifrar su sistema de archivos, marque la casilla Cifrar . Luego, aparecerán dos campos para obtener escritura; escriba su contraseña de cifrado dos veces para confirmarla. Para agregar una etiqueta para esta partición, introdúzcala en el campo Etiqueta FS .



El gestor de arranque GRUB de Lubuntu no permite crear una partición cifrada / sin una partición /boot independiente sin cifrar. Si crea una partición /home independiente, guardará todos sus documentos, fotos y vídeos, así que asegúrese de que tenga suficiente espacio para sus archivos personales. No todas las particiones creadas estarán en la misma unidad, por ejemplo, puede colocar /home en su propia unidad para obtener suficiente espacio y luego colocar / desde donde se iniciarán todo su sistema y programas. Un buen ejemplo de una forma sensata de usar el particionado manual es si tiene un disco duro y un disco de

estado sólido en una computadora: coloque la partición / en el disco duro para iniciar los programas más rápido y /home en el disco duro para tener más espacio para sus archivos.

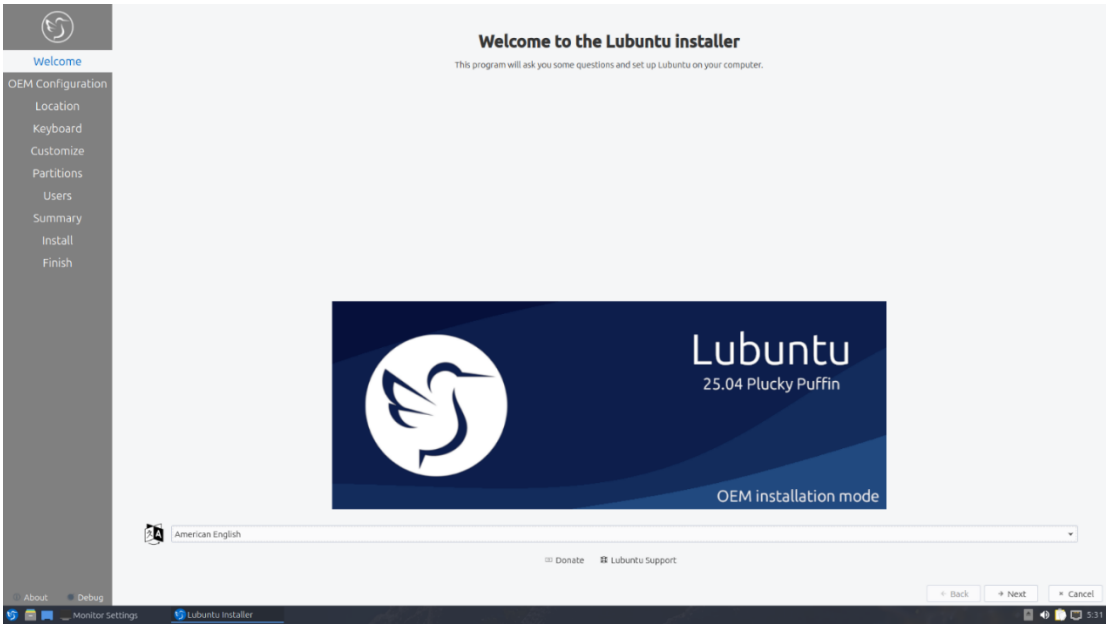
Para revertir todos los cambios al estado anterior, pulse el botón "Revertir todos los cambios" en la esquina superior derecha. Para cambiar el dispositivo desde el que arrancará su ordenador, deberá usar el menú desplegable "Instalar cargador de arranque" para seleccionar el disco desde el que arrancar.

En el centro de la ventana de particionado manual se muestra el nombre de la partición. El tipo de sistema de archivos se muestra en la columna " Sistema de archivos" . Para ver dónde está montada la partición, consulte la columna "Punto de montaje" . El tamaño del sistema se muestra en la columna " Tamaño" .

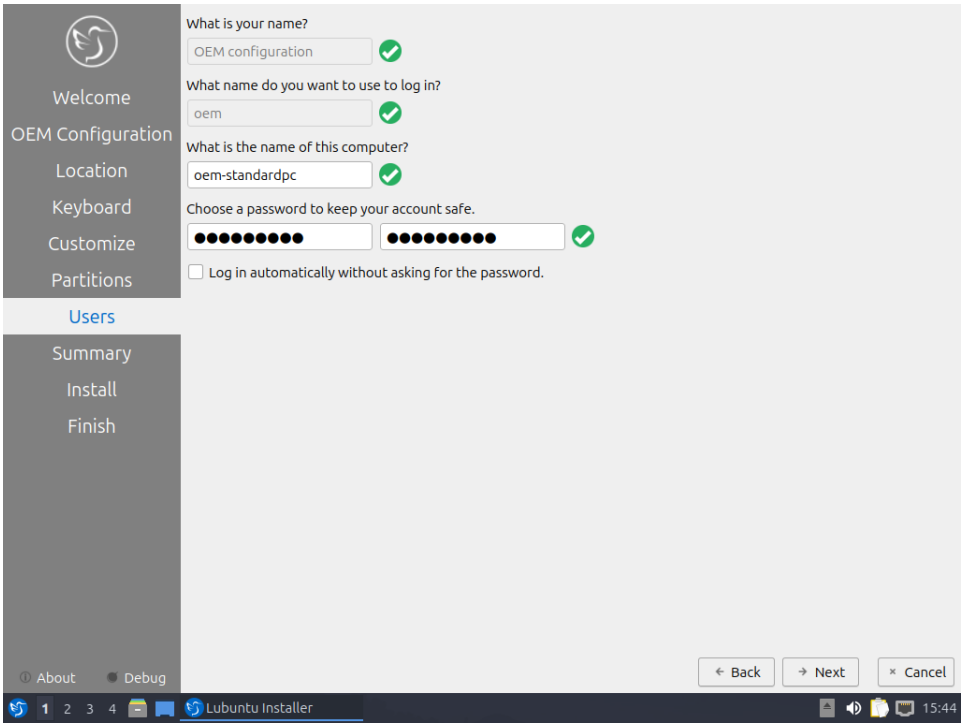
En la parte superior de la ventana de particionado manual, se muestra una barra visual que muestra el tamaño de las particiones. Debajo, se muestra cada partición de la unidad con su color en el gráfico de barras y su tamaño.

Instalaciones OEM

Para instalar Lubuntu en modo OEM y distribuirlo a otra persona en la pantalla de inicio de GRUB, seleccione "Instalación OEM" (para fabricantes) o, en el menú del panel , seleccione "Herramientas del sistema ▸ Instalar Lubuntu 24.10 (modo OEM)" . Se le preguntará si está seguro de que desea instalar en modo OEM. Para hacerlo, pulse el botón "Sí" . El instalador mostrará una pantalla de bienvenida indicando que está en modo OEM. Para cambiar el idioma de instalación, utilice la barra desplegable en "Modo de instalación OEM" . Para ir al siguiente paso de la instalación, pulse el botón "Siguiente" .



Para establecer un nombre para este sistema preinstalado que está instalando para enviárselo a alguien, introdúzcalo en el campo Lote . Para pasar a la siguiente parte, pulse Siguiente . Para volver a la bienvenida, pulse el botón Atrás . Las secciones Ubicación , Teclado , Personalizar y Particiones son las mismas que en una instalación normal. Para la instalación OEM, no podrá crear un nombre de usuario, ya que el usuario final al que se lo envíe lo utilizará. Para elegir un nombre de host, introdúzcalo en el campo ¿Cuál es el nombre de esta computadora?. A continuación, introducirá una contraseña dos veces en el campo Elija una contraseña para mantener su cuenta segura . Para que su instalación OEM inicie sesión automáticamente, marque la casilla Iniciar sesión automáticamente sin solicitar la contraseña . Para pasar a la siguiente parte de la instalación, pulse el botón Siguiente .

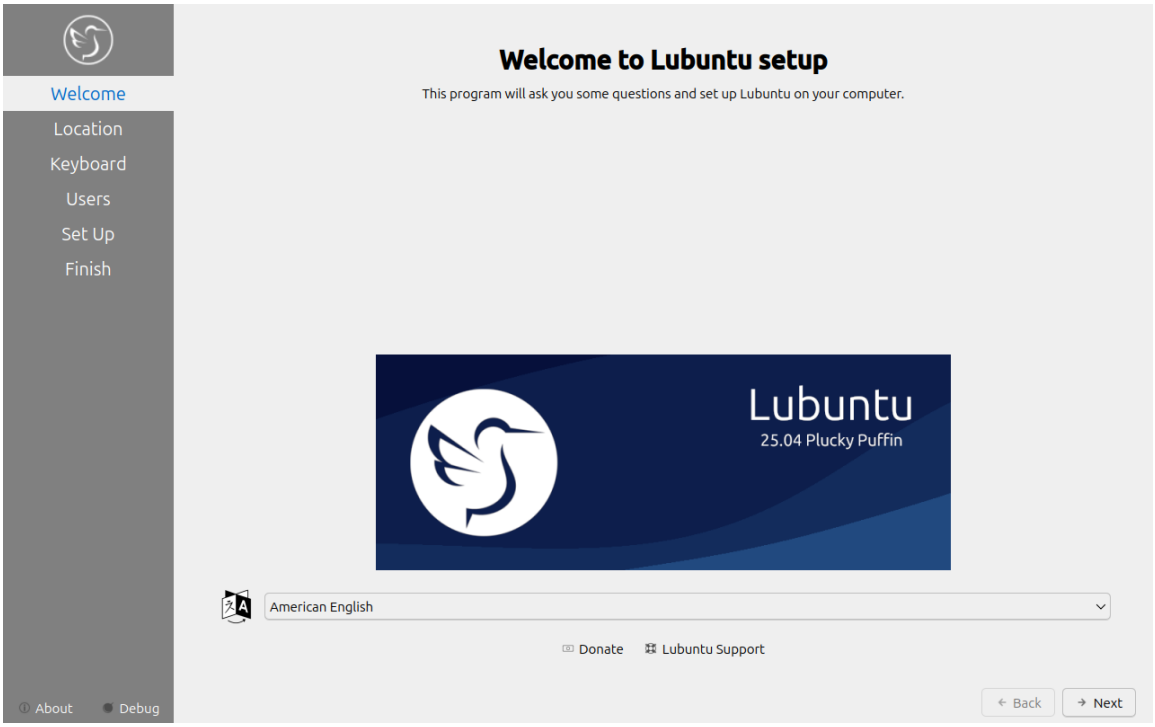


A continuación, una pantalla de resumen muestra la configuración de sus usuarios. Para comenzar la instalación, pulse el botón Instalar y, para confirmar que es la correcta, pulse el botón Instalar ahora .

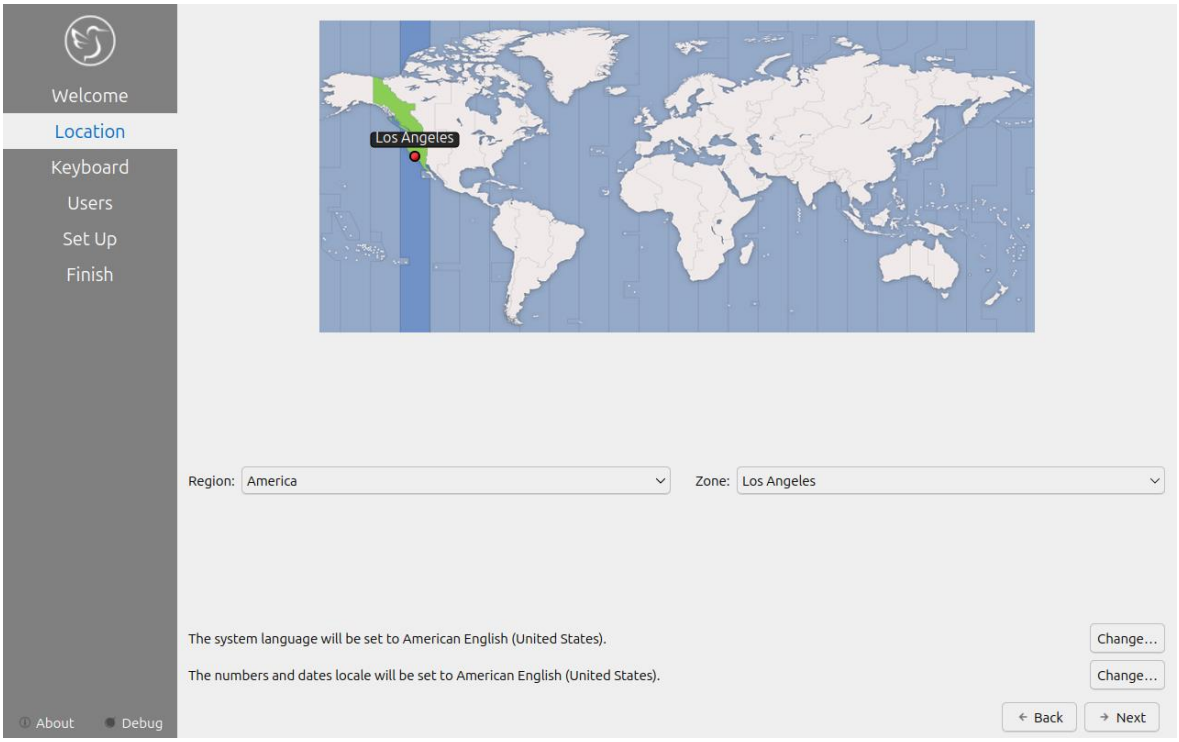
Configuración de usuario OEM

Para configurar la configuración de usuario para un sistema OEM preinstalado , seleccione Herramientas del sistema ▸ Finalizar la preparación del OEM . Se le preguntará si desea finalizar la configuración del OEM y se ejecutará un asistente en el siguiente arranque. Pulse Sí para confirmar. Deberá reiniciar para ejecutar la configuración de usuario.

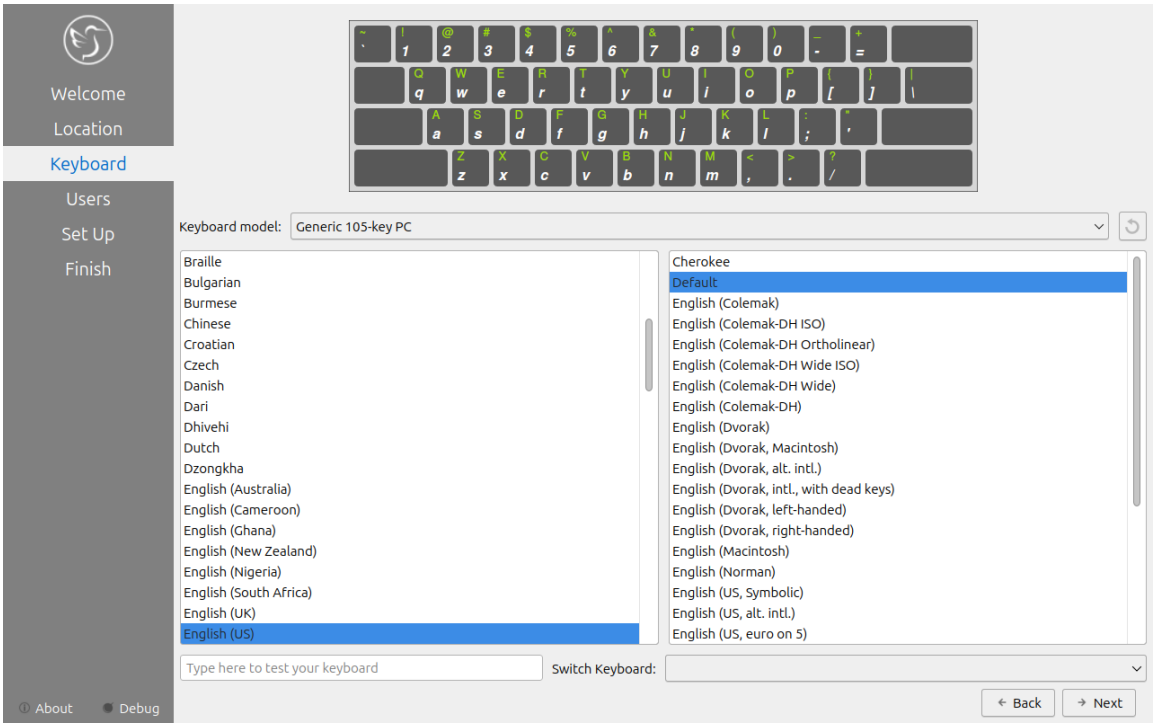
La primera sección de configuración del usuario le preguntará en qué idioma desea continuar al final. Para pasar a la siguiente parte de la configuración de Lubuntu, pulse el botón Siguiente .



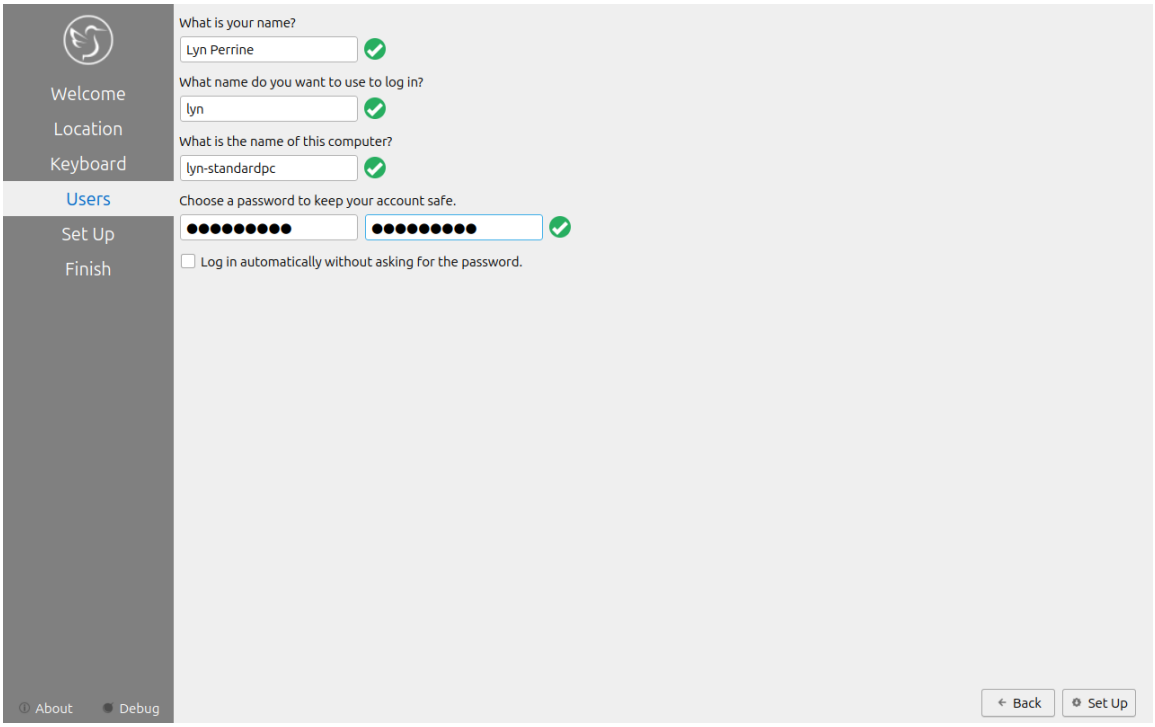
A continuación, deberá seleccionar su zona horaria en los campos "Región" y "Zona" , con un mapa del mundo encima que le indicará la hora establecida. Para configurar el idioma de inicio del sistema, pulse el botón "Cambiar " junto al campo "El idioma del sistema se establecerá en ". Para cambiar el formato de fecha y número, pulse el botón " Cambiar " junto al campo "La configuración regional de números y fechas se establecerá en ". Para ir a la siguiente parte de la configuración de Lubuntu, pulse el botón " Siguiente" .



El siguiente paso de la configuración será seleccionar la distribución del teclado. Para cambiar el modelo de teclado, cambie el campo "Modelo de teclado" . La columna izquierda define el idioma del teclado, mientras que la columna derecha selecciona la distribución. En la esquina inferior izquierda, escriba para asegurarse de que lo que cree escribir coincida con lo que se está escribiendo. Para configurar una tecla y cambiar la distribución del teclado, seleccione una en el campo "Cambiar teclado" . Para configurar los usuarios, pulse el botón " Siguiente" .



A continuación, configure los usuarios de su computadora. Ingrese su nombre en el campo "¿Cuál es su nombre?" . Ingrese su nombre de usuario en el campo de inicio de sesión . Ingrese su nombre de host en " ¿Cuál es el nombre de su computadora?" . Escriba su contraseña dos veces para asegurarse de escribirla correctamente en los campos bajo " Elija una contraseña para proteger su cuenta" . Para iniciar sesión automáticamente, marque la casilla "Iniciar sesión automáticamente sin solicitar la contraseña ".



Para finalizar la creación de usuarios y la configuración del teclado e idioma, presione el botón "Configurar" . Se le pedirá confirmación, ya que no es posible deshacer este paso. Para continuar con la configuración, presione " Configurar ahora" . Los usuarios se configurarán con una presentación de diapositivas. Después, podrá reiniciar el sistema y, una vez reiniciado, podrá iniciar sesión y usar Lubuntu.

Anexo C. Instalación de OMNET++

El presente anexo describe de manera resumida el proceso de instalación y configuración del entorno de simulación OMNeT++, herramienta empleada para el desarrollo

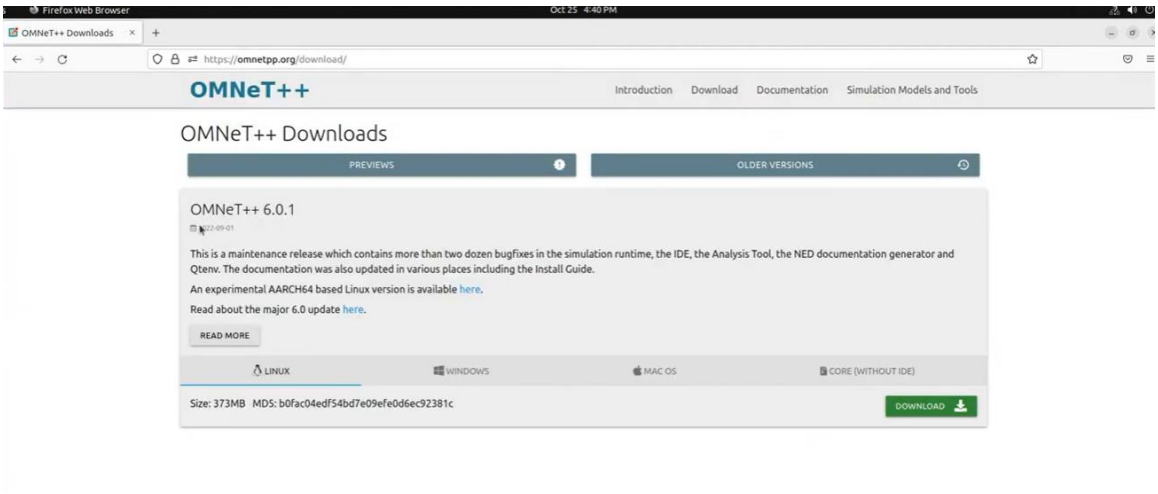
y validación de los escenarios experimentales del presente trabajo de titulación. OMNeT++ es un simulador orientado a eventos discretos ampliamente utilizado en investigación de redes de comunicación, especialmente en el análisis de protocolos, modelado de topologías y evaluación de desempeño en entornos controlados.

La correcta instalación del entorno de simulación constituye un requisito fundamental para garantizar la reproducibilidad de los experimentos realizados, así como la adecuada integración de librerías y frameworks complementarios utilizados en este estudio (por ejemplo, INET y módulos específicos para comunicación subacuática). En este anexo se detallan los pasos seguidos para la instalación en sistema operativo GNU/Linux, incluyendo la preparación del entorno, dependencias requeridas, compilación y verificación del correcto funcionamiento del simulador.

El objetivo de esta sección es proporcionar una guía técnica clara que permita replicar el entorno experimental bajo las mismas condiciones configuradas en esta investigación.

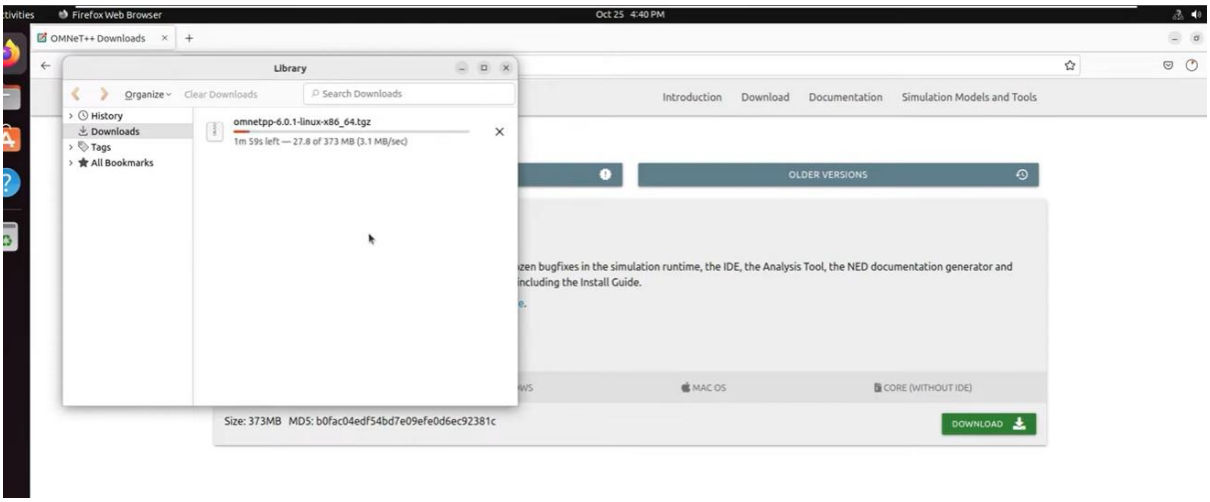
En la siguiente imagen se muestra el acceso al portal oficial de descargas de OMNeT++ (https://omnetpp.org/download/), desde donde se obtiene la versión utilizada en esta investigación.

Para el desarrollo de los escenarios experimentales se seleccionó la versión OMNeT++ 6.0.1, correspondiente a una versión estable de mantenimiento. En el portal se eligió la opción de descarga para sistema operativo Linux, considerando que el entorno experimental fue configurado bajo una distribución GNU/Linux.



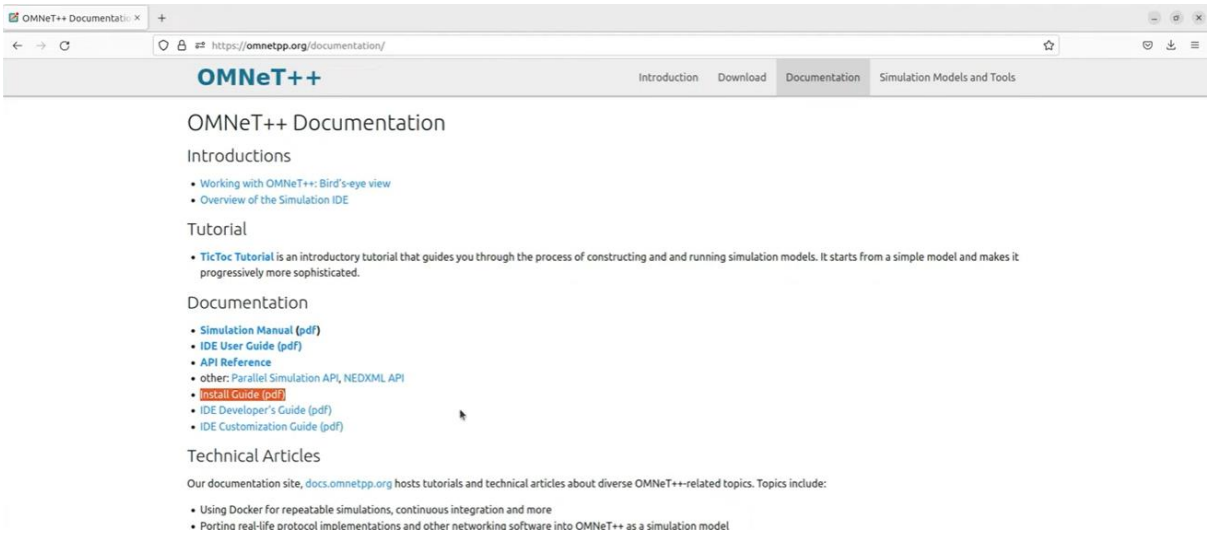
En la siguiente imagen se observa el proceso de descarga del archivo omnetpp-6.0.1-linux-x86_64.tgz, correspondiente al paquete de instalación para Linux.

El archivo comprimido (.tgz) contiene el entorno completo del simulador y será utilizado en los siguientes pasos para su descompresión y compilación en el sistema operativo.

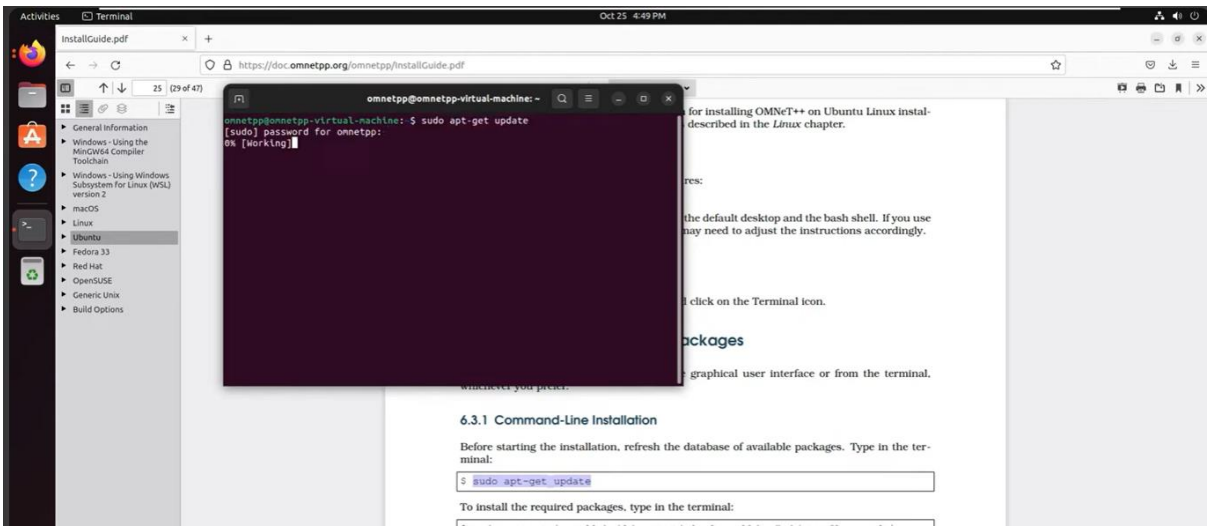


A continuación, se muestra la sección **Documentation** del sitio oficial de OMNeT++.

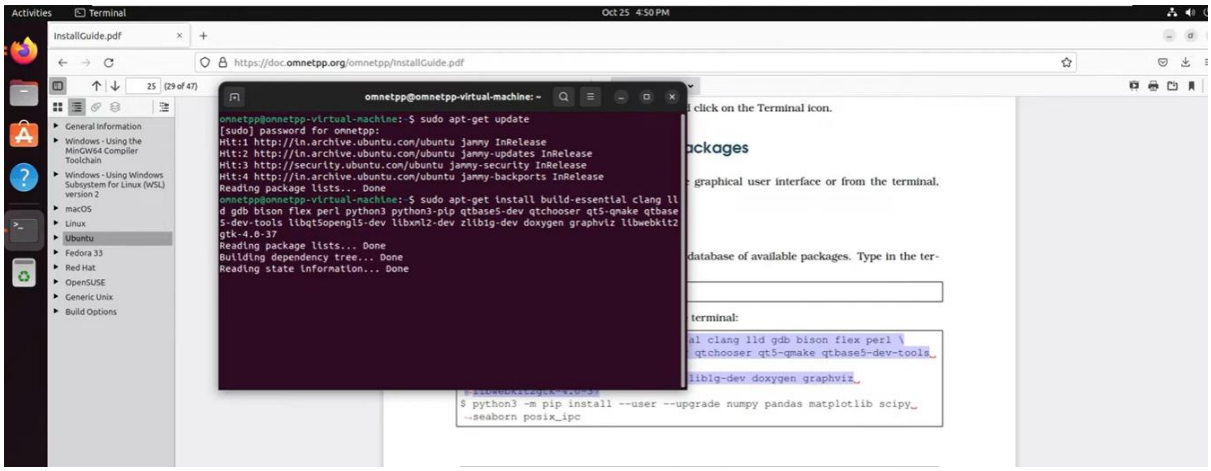
En esta sección se consultó la **Install Guide**, utilizada como referencia para identificar dependencias y pasos de instalación en Linux.



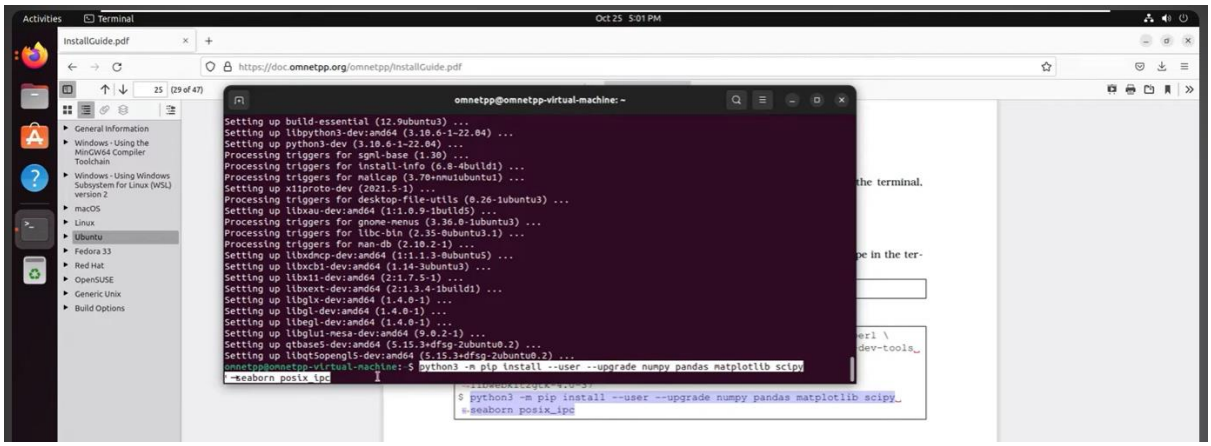
A continuación, se muestra la actualización de los repositorios del sistema mediante el comando sudo apt-get update en la terminal de Linux, paso necesario previo a la instalación de las dependencias requeridas por OMNeT++.



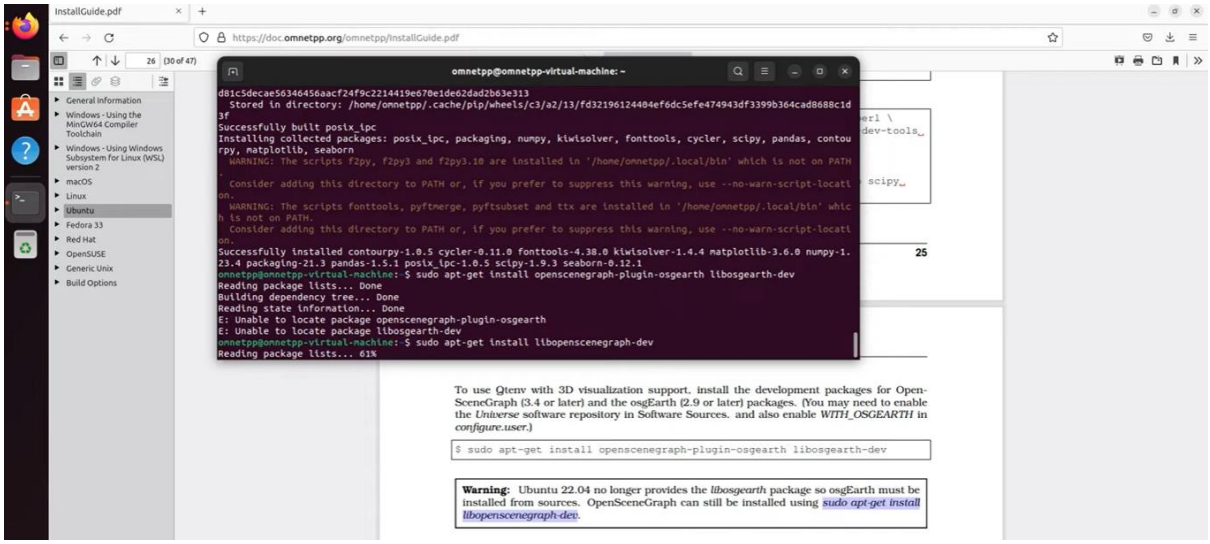
A continuación, se muestra la instalación de las dependencias necesarias para OMNeT++ mediante el comando sudo apt-get install build-essential clang lldb gdb bison flex perl python3 qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools libxml2-dev zlib1g-dev doxygen graphviz, garantizando el correcto proceso de compilación del simulador en entorno Linux.



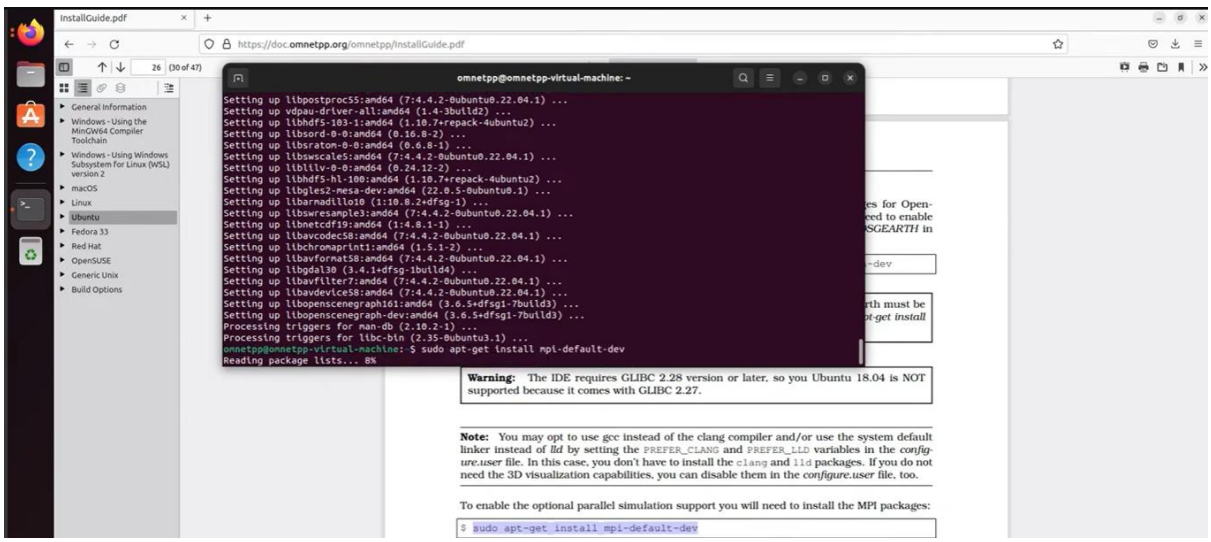
Se muestra la finalización del proceso de instalación de las dependencias del sistema, confirmando que los paquetes necesarios fueron configurados correctamente para proceder con la compilación de OMNeT++.



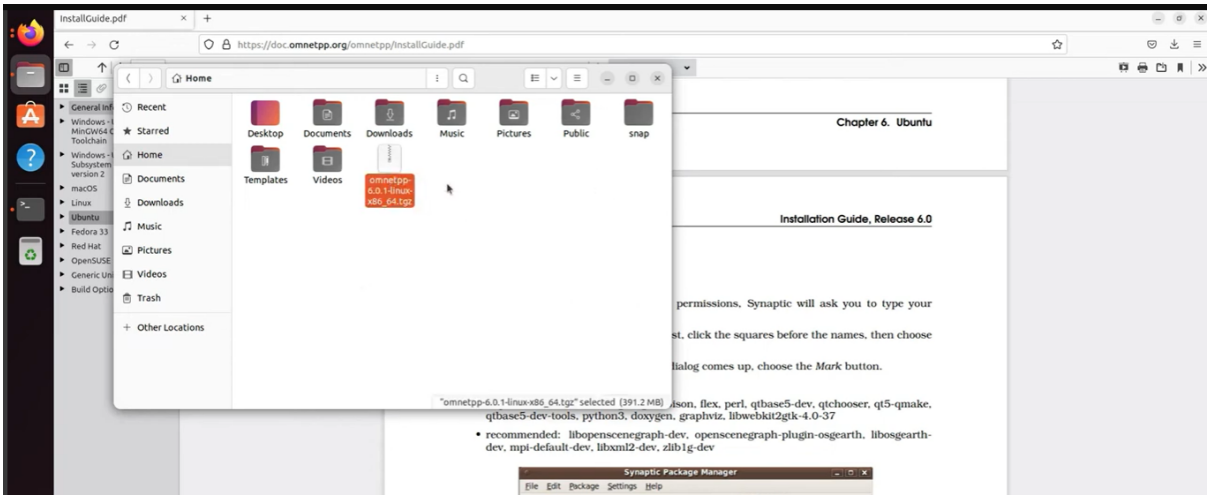
La instalación adicional del paquete libopenscenegraph-dev, requerido para habilitar soporte de visualización 3D en OMNeT++, ejecutando el comando correspondiente en la terminal.



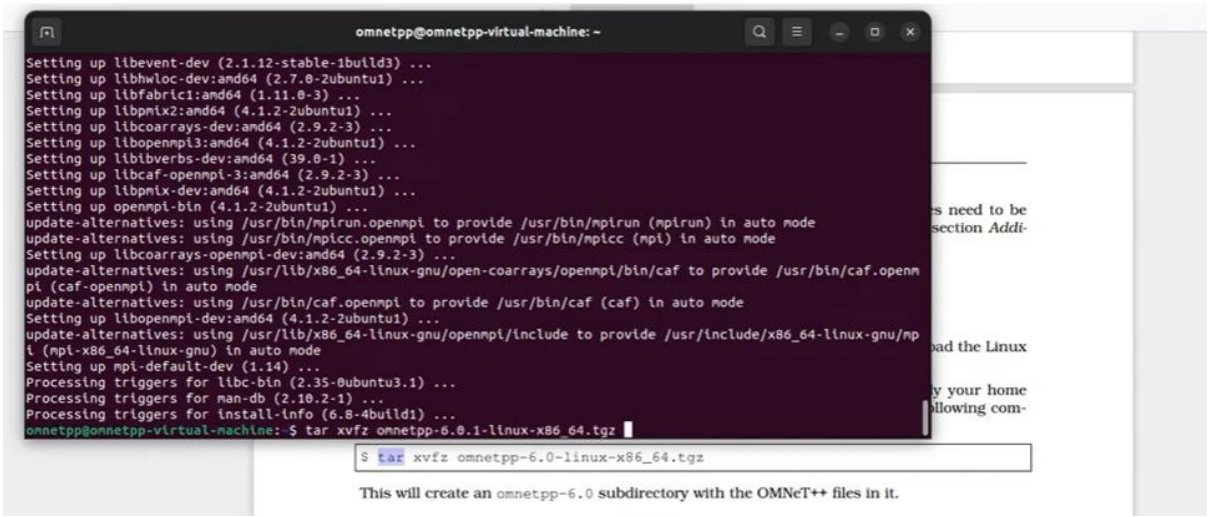
A continuación, se muestra la instalación del paquete mpi-default-dev, necesario para habilitar el soporte de simulación paralela en OMNeT++, completando así las dependencias opcionales del entorno.



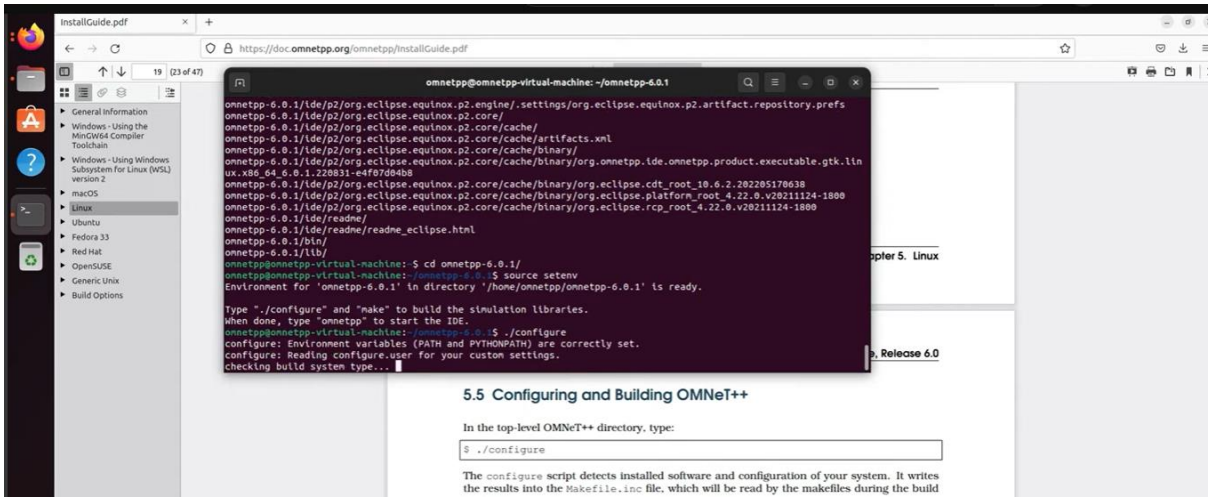
A continuación, se muestra el archivo comprimido omnetpp-6.0.1-linux-x86_64.tgz descargado en el sistema, el cual será utilizado en el siguiente paso para su descompresión e instalación del entorno de simulación.



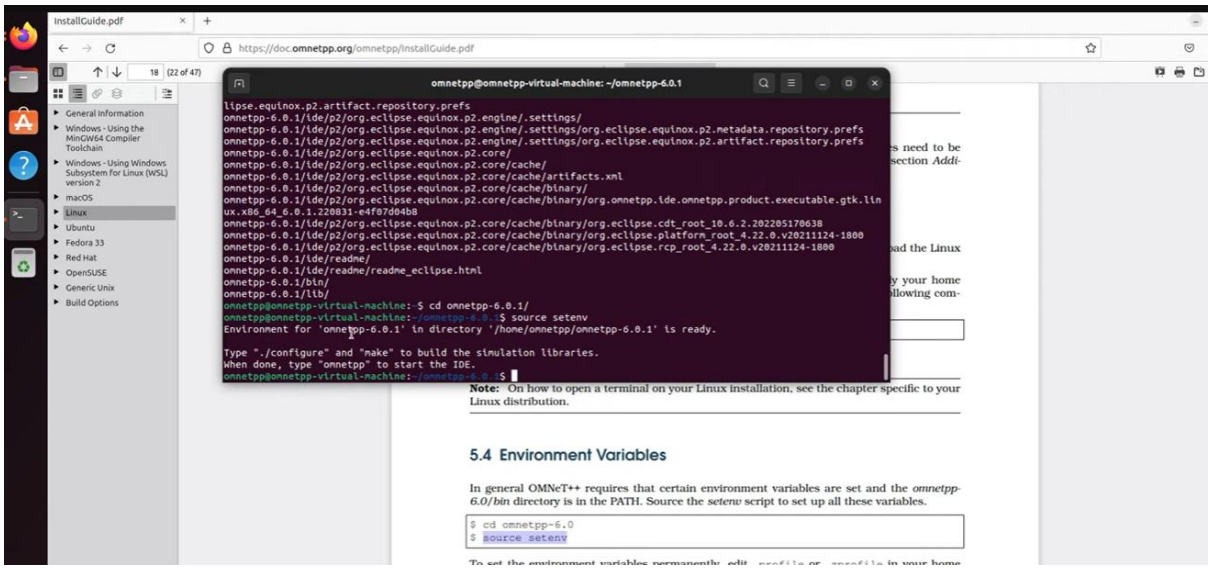
Se muestra la descompresión del archivo omnetpp-6.0.1-linux-x86_64.tgz mediante el comando tar xvfz, generando el directorio de instalación del simulador en el sistema.



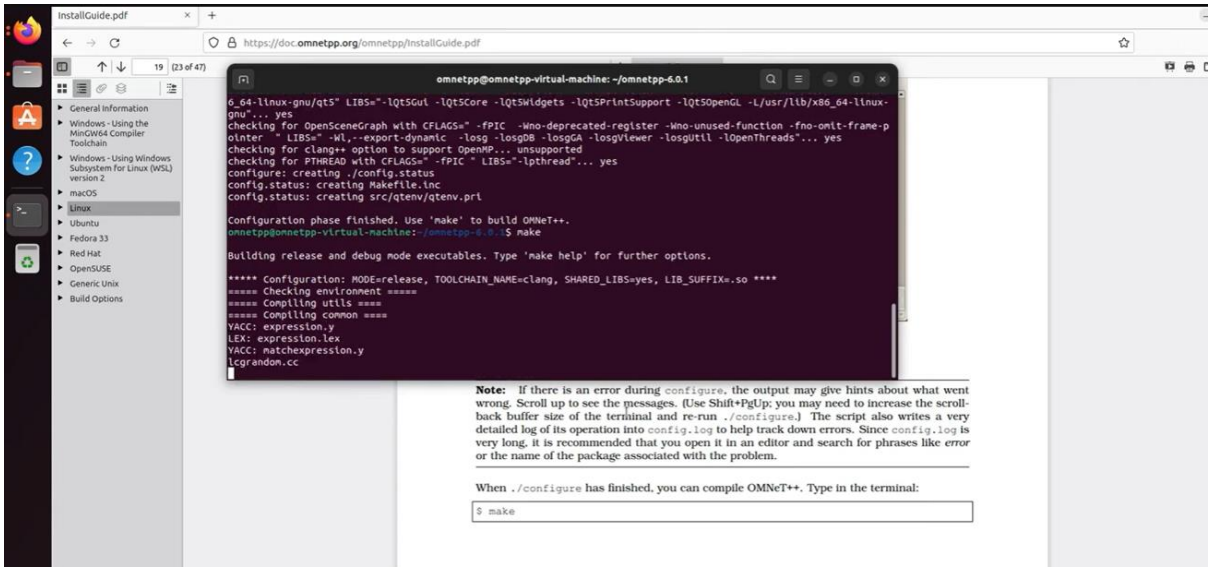
Seguidamente, se muestra el acceso al directorio omnetpp-6.0.1 y la ejecución del script source setenv, comando que configura las variables de entorno necesarias para preparar el sistema antes del proceso de compilación.



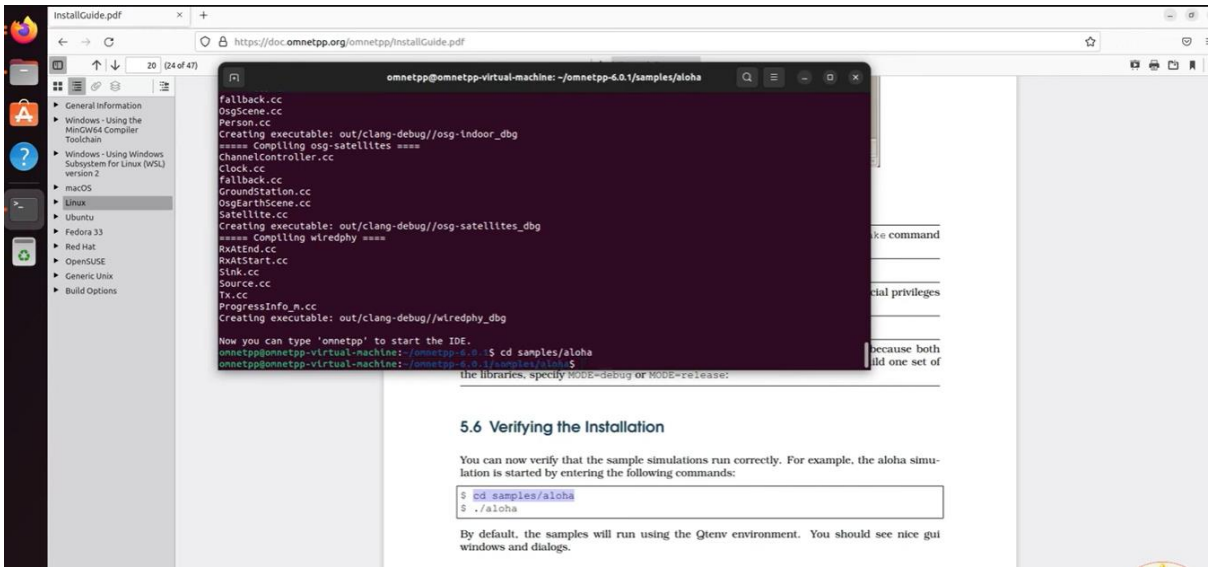
Posteriormente, se ejecuta el comando ./configure dentro del directorio de OMNeT++, permitiendo verificar las dependencias instaladas y preparar los archivos necesarios para la compilación del simulador.



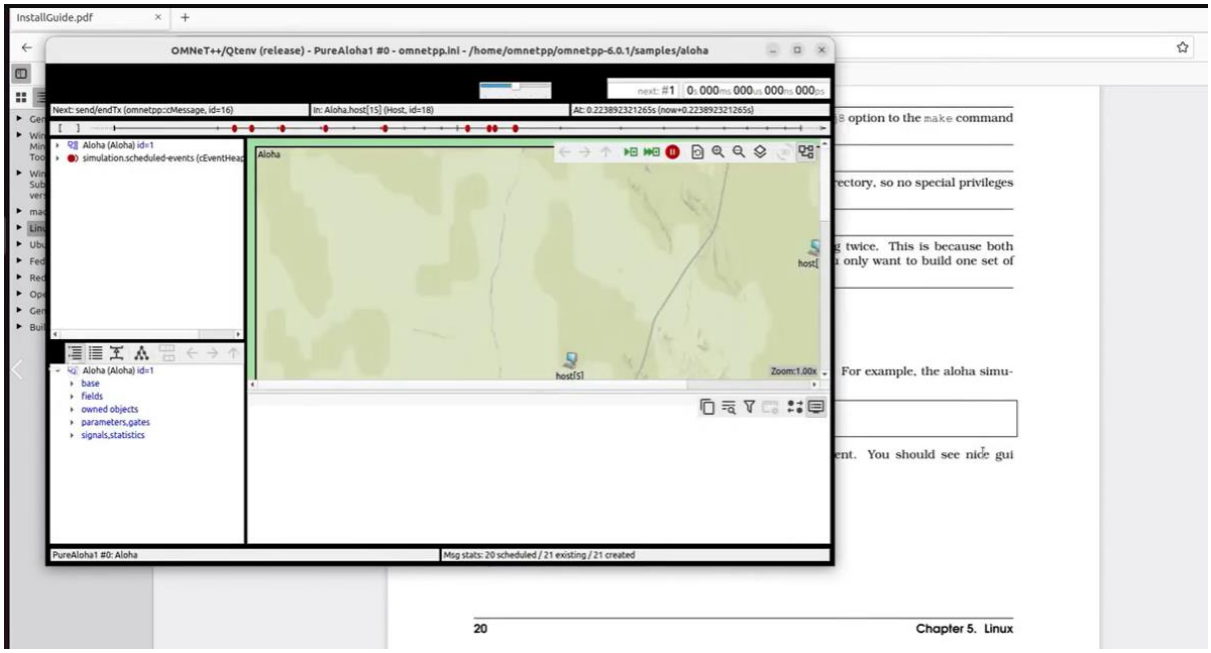
Finalmente, se ejecuta el comando make, iniciando el proceso de compilación del entorno OMNeT++, donde se generan los ejecutables y librerías necesarias para el funcionamiento del simulador.



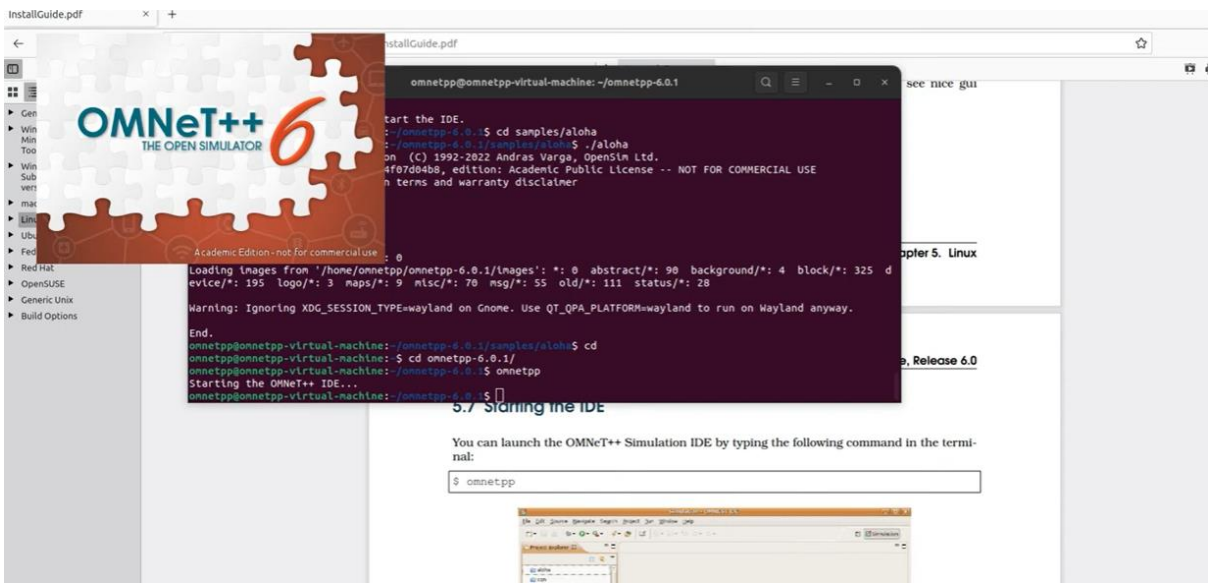
Finalmente, se muestra el acceso al directorio samples/aloha, utilizado para verificar el correcto funcionamiento del entorno mediante la ejecución de un ejemplo de simulación incluido por defecto en OMNeT++.



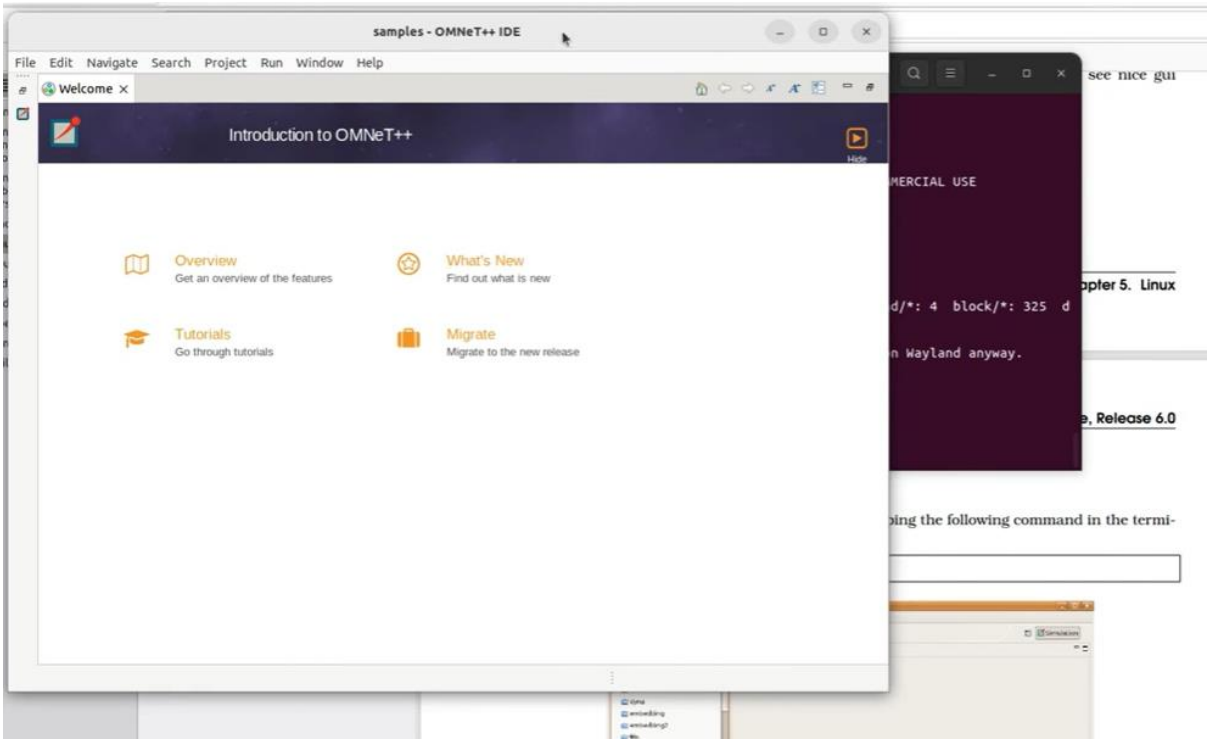
Por último, se observa la ejecución correcta del ejemplo Aloha dentro del entorno gráfico de OMNeT++, confirmando que la instalación y compilación del simulador se realizaron exitosamente.



Finalmente, se muestra la ejecución del comando omnetpp, iniciando el entorno gráfico del simulador y confirmando que el IDE se abre correctamente tras completar el proceso de instalación.



Por último, se muestra la interfaz principal del IDE de OMNeT++, confirmando que el entorno gráfico se ha iniciado correctamente y que el simulador se encuentra listo para la creación y ejecución de proyectos.


samples - OMNeT++ IDE
File Edit Navigate Search Project Run Window Help
Welcome
Introduction to OMNeT++
Overview
Get an overview of the features
What's New
Find out what is new
Tutorials
Go through tutorials
Migrate
Migrate to the new release
see nice gui
MERCIAL USE
pter 5. Linux
n Wayland anyway.
e, Release 6.0
ing the following command in the termi-