ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

ANÁLISIS DEL PARADIGMA DE SEGURIDAD DEFINIDO EN EL ESTÁNDAR IEEE 802.15.6, MEDIANTE UN ESCENARIO DE PRUEBAS

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN ELECTRÓNICA Y REDES DE INFORMACIÓN

ANA ESTHELA SANGUIL LLANGARI

ana.sanguil@epn.edu.ec

DIRECTOR: MSc. PABLO WILLIAM HIDALGO LASCANO

pablo.hidalgo@epn.edu.ec

Quito, enero 2020

AVAL

Certifico que el presente trabajo fue desarrollado por Ana Esthela Sanguil Llangari, bajo mi supervisión.

_____

MSc. PABLO WILLIAM HIDALGO LASCANO
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo Ana Esthela Sanguil Llangari, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

_____

ANA ESTHELA SANGUIL LLANGARI

DEDICATORIA

Le dedico todo mi trabajo en primer lugar a Dios, por darme salud y vida para permitirme alcanzar este objetivo planteado hace varios años.

A mis padres, porque ellos son mis más grandes educadores y ejemplo más grande de esfuerzo constante. Este logro es mas de ellos que mío.

A mis hermanos, Nancy, Pedro, Iván y Lucia; quienes siempre me dieron palabras de aliento para no rendirme jamás.

A mi hermano José, por trabajar tantos años junto a mis padres para bridarme la posibilidad de estudiar y formarme como una persona de bien.

A mi amado novio Jonathan, parte importante de mi vida y apoyo constante para lograr mis objetivos.

AGRADECIMIENTO

Mis primeras líneas de agradecimiento se las dedico a Dios por cada experiencia vivida, las cuales se han convertido en lecciones de vida.

Agradezco a todos mis profesores, a mis padres y hermanos que en el transcurso del tiempo han formado mi carácter y han transmitido su sabiduría.

A mi profesor titular, MSc. Pablo Hidalgo por encaminar este trabajo brindándome su apoyo; siendo más que un educador un verdadero maestro.

Por último, agradezco a mis mejores amigos que siempre han puesto palabras de ánimo y apoyo en mi mente y corazón, esto también se los debo a ustedes.

ÍNDICE DE CONTENIDO

RESUMEN

El auge de nuevas tecnologías permite al ser humano mejorar su calidad de vida mediante la utilización de dispositivos que permiten controlar aspectos cotidianos. Las WBAN (*Wireless Body Area Network*) se han creado con el fin de cubrir las necesidades tecnológicas que corresponde a un área delimitada como es la del cuerpo humano. Existe un amplio rango de actividades donde las WBAN pueden ser aplicadas y cuyos beneficios pueden ser aprovechados.

Sin embargo, las WBAN también se encuentran expuestas a riesgos de seguridad que deben ser solventados, ya que como toda red de comunicación requiere que se garantice la privacidad de la información que se comparte. Es por ello que, el actual trabajo de titulación, busca evidenciar la seguridad ofrecida por el paradigma de seguridad establecido en el estándar IEEE 802.15.6 mediante pruebas de concepto.

El primer capítulo realiza una breve introducción a las WBAN, características del estándar IEEE 802.15.6, su estructura de seguridad y mecanismos del mismo. En el segundo capítulo se indican el diseño y la implementación de los procesos presentes en el paradigma de seguridad como son Asociación, Creación PTK (*Pairwise Temporal Key*) y Seguridad del Mensaje. El tercer capítulo muestra los resultados obtenidos al implementar el paradigma y también las pruebas de concepto establecidas para analizar su seguridad. En el cuarto capítulo se presentan las conclusiones, a las cual se ha llegado después realizar un análisis a los resultados obtenidos; además, también se presentan en este mismo capítulo las recomendaciones inherentes al tema. Por último, en la sección de Anexos se encuentran tablas con los valores de las tramas utilizadas en los procesos, diagramas de secuencia, el código desarrollado para lograr los objetivos planteados, diagramas de clase y el estándar IEEE 802.15.6 en el cual se basa el estudio de este trabajo de titulación.

PALABRAS CLAVE: WBAN, IEEE 802.15.6, Pairwise Temporal Key, CMAC, CCM.

ABSTRACT

The rise of new technologies allows human improve their quality of life by implementing devices that allow them to control everyday aspects. The WBANs (Wireless Body Area Network) been created in order to cover the technological needs that correspond to a delimited area as referred to the human body. There is a wide range of activities where WBANs can be applied and whose benefits can be exploited.

However, WBANs are also expose to security risks that should been solved, since, like any communication network, require that the privacy of the shared information be guaranteed. That is why the current degree work seeks to show the security offered by the security paradigm established in the IEEE 802.15.6 standard through proof of concept.

The first chapter gives a brief introduction to the WBAN, characteristics of the IEEE 802.15.6 standard, its security structure and its mechanisms. The second chapter indicates the design and implementation of the processes present in the security paradigm such as Association, PTK Creation and Message Security. The third chapter shows the results obtained by implementing the paradigm and the proofs of concept established to analyze its safety. In the fourth chapter the conclusions been presented, which has been reached afterwards to carry out an analysis of the results obtained, in addition, the recommendations inherent to the subject are also presented in this same chapter. Finally, in the Annexes section there are tables with the values of the frames used in the processes, sequence diagrams, the code developed to achieve the stated objectives, class diagrams and the IEEE 802.15.6 standard on which the study of this degree work been based.

KEYWORDS: WBAN, IEEE 802.15.6, Pairwise Temporal Key, CMAC, CCM.

1. INTRODUCCIÓN

Las Redes Inalámbricas de Área Corporal o WBAN permiten obtener información del cuerpo humano mediante un conjunto de dispositivos de corto alcance y baja potencia que pueden ser colocados dentro o fuera del mismo, con diferentes finalidades, para después integrarse a otros niveles de red.

Se asume que una WBAN está en la capacidad de brindar altos niveles de seguridad, puesto que la información que aquí se maneja está catalogada como sensible en base al daño personal, invasión de privacidad y a una posible pérdida de derechos [1], que puede ser ocasionada al individuo por la utilización de esta tecnología. Sin embargo, no existen muchos análisis de seguridad que muestren un estudio, corroborando o contradiciendo la seguridad en este tipo de red.

Lo mencionado anteriormente causa incertidumbre en la utilización de las WBAN para la transferencia de información en diferentes áreas de aplicación, ya que se podrían generar graves problemas, como, por ejemplo: en el área de la salud, la obtención o alteración de datos del paciente puede ocasionar un mal diagnóstico y causar serios perjuicios, e incluso hasta su deceso.

Las WBAN se encuentran normalizadas en el estándar IEEE 802.15.6 [2], donde se define un paradigma de seguridad para la comunicación segura entre Nodo y *Hub*. El paradigma se encuentra estructurado con tres procesos fundamentales, que permiten que el intercambio de información se realice entre las entidades autorizadas. Si al menos uno de los procesos dispuestos en el paradigma de seguridad es susceptible de ser vulnerado, entonces la comunicación se ve comprometida. Así lo explica [3], donde se realiza un análisis teórico sobre las vulnerabilidades existentes en los protocolos criptográficos de [2]. Otro análisis [4], determina, en términos de probabilidad, que el modo CCM (*Counter with CBC-MAC*[1]) [6], utilizado en uno de los procesos del paradigma de seguridad de [2], presenta problemas de seguridad en la autenticación y privacidad.

Por lo anterior, el presente trabajo de titulación plantea realizar un análisis de seguridad del paradigma establecido en [2] para una comunicación segura, específicamente analizando dos de los procesos aquí definidos, los cuales son: Creación PTK y Seguridad del Mensaje, con la finalidad de dar a conocer el nivel de seguridad que provee el estándar IEEE 802.15.6.

[1] *CBC-MAC* método que provee integridad a los mensajes, utiliza un esquema de cifrado de bloque como los provistos por DES y AES [5].

Este trabajo de titulación, ayudará a evitar que se siga asumiendo que el estándar que definen las WBAN [2], proporciona un nivel de seguridad óptimo y que, de no ser el caso, se generarían problemas en las diferentes áreas de aplicación y consecuentemente un rechazo y desuso del estándar.

1.1. OBJETIVOS

El objetivo general de este Proyecto Técnico es:

- Analizar el paradigma de seguridad definido en el Estándar IEEE 802.15.6, mediante un escenario de pruebas.

Los objetivos específicos de este Proyecto Técnico son:

- Describir los procesos presentes en las etapas dos y tres del paradigma de seguridad definido en el Estándar IEEE 802.15.6.
- Diseñar el software para la evaluación del paradigma de seguridad de IEEE 802.15.6.
- Implementar los componentes software en un escenario de comunicación para emular una comunicación básica WBAN.
- Analizar los resultados a partir de las pruebas realizadas.

1.2. ALCANCE

El presente proyecto analiza el paradigma de seguridad establecido en [2], que se lleva a cabo en una comunicación segura de nivel 2, entre Nodo y *Hub* dentro de una WBAN. Toma como base el proyecto de titulación "Evaluación de Vulnerabilidades del protocolo *Display Authenticated Association*, definido en el Estándar IEEE 802.15.6, mediante la implementación de un escenario de pruebas" [7], el cual desarrolla la primera fase del paradigma: Asociación. Para el análisis de las vulnerabilidades presentes en el mismo, se procede a desarrollar y analizar las etapas dos y tres que prosiguen en la estructura del paradigma de seguridad. En la segunda etapa se despliega el proceso de Creación PTK, que hace uso del modo de operación de cifrado de bloque para autenticación CMAC (*Cipher-based Message Authentication Code*) publicado por el NIST SP 800-38b [7], en conjunto con un cifrado de bloque de clave simétrica como el AES (*Advanced Encryption Standard*) disponible en FIPS Pub 197 [8], utilizados para el intercambio seguro de tramas.

La tercera etapa, está orientada a la autenticación y confidencialidad de los datos; para lograr este objetivo el estándar IEEE 802.15.6 establece el uso del modo de operación de cifrado de bloque CCM definido en el RFC 3610 [6], que además también usa AES como su función de cifrado. Por lo tanto, es importante realizar una revisión bibliográfica acerca de las herramientas empleadas en las distintas etapas del paradigma de seguridad con la finalidad de entender su funcionamiento e implementarlas correctamente para su posterior análisis.

Una vez realizada la revisión bibliográfica y haciendo uso del lenguaje C# se lleva a cabo la programación del paradigma de seguridad que tiene lugar entre dos entidades, Nodo y *Hub*. En la etapa de Asociación se hace uso del código desarrollado en [7] escrito en lenguaje Java para ser codificado a C# y posteriormente se implementan las etapas de las siguientes dos fases. Una vez que se ha plasmado el paradigma de seguridad en lenguaje C#, se trata de determinar el nivel de seguridad que pueden proporcionar las etapas dos y tres, analizando sus fortalezas y debilidades mediante diferentes pruebas de concepto que son determinadas durante el desarrollo del proyecto. La Figura 1.1 representa el escenario de comunicación que se pretende implementar.

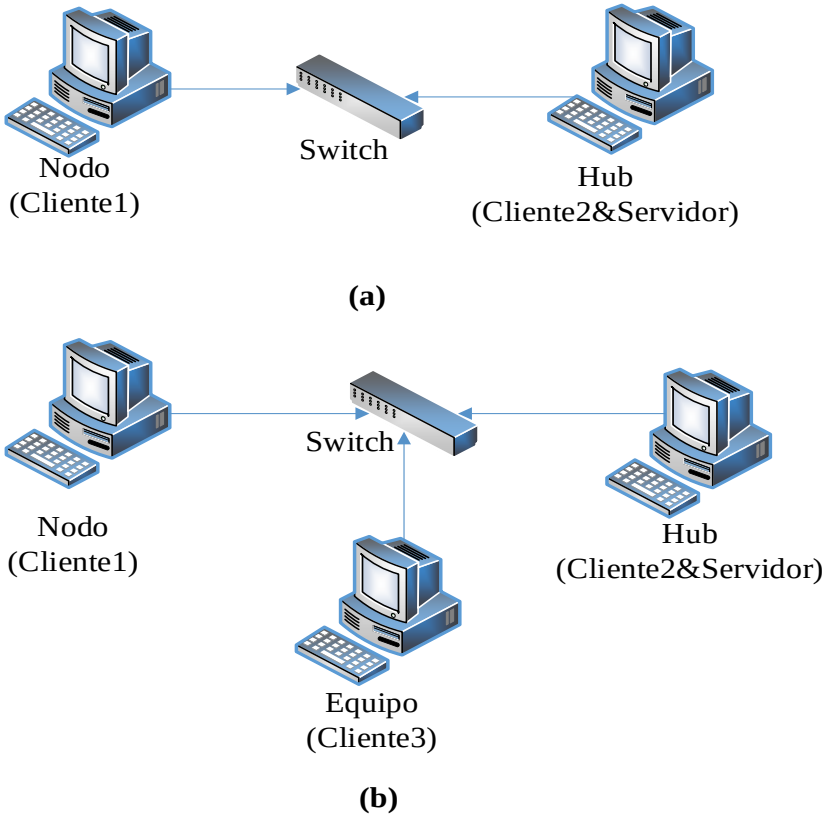


Figura 1.1. Escenario de comunicación entre entidades autorizadas. (b) Escenario de comunicación para pruebas.

La Figura 1.2 muestra el diagrama de componentes software, que permite identificar el bosquejo inicial de la comunicación para el desarrollo del paradigma de seguridad entre Nodo y *Hub*.

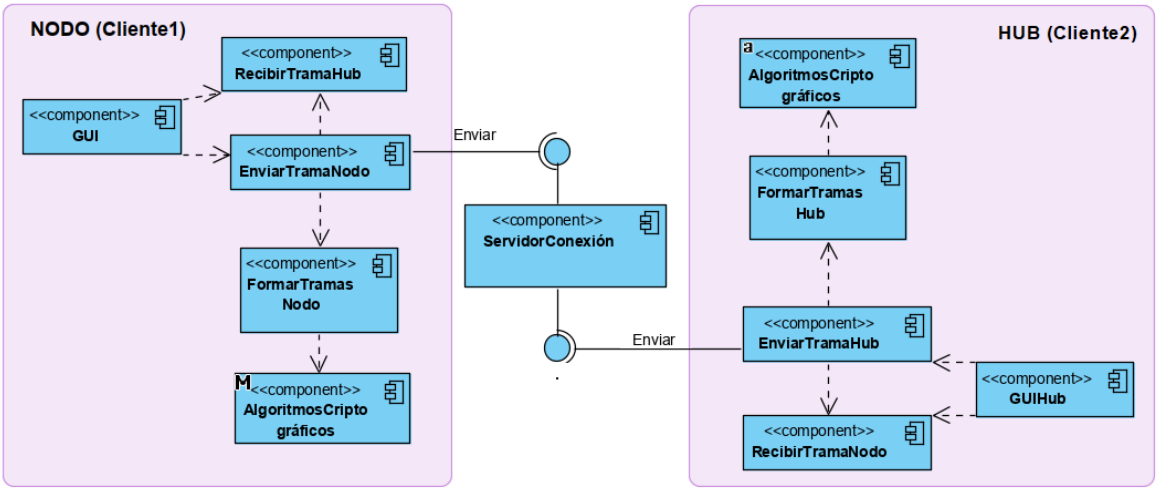


Figura 1.2. Diagrama de componentes para el desarrollo del paradigma de seguridad.

Es importante mencionar que en el presente trabajo no se implementarán protocolos más allá de los mencionados, es decir que no se creará un ambiente WBAN real con todas las características de la capa PHY (*Physical*) y subcapa MAC (*Medium Access Control*). Además, para la emulación de las tramas *Beacon,* Solicitud de Conexión y Asignación de Conexión e *I-Ack* se hace uso de los campos importantes de cada una de las tramas mencionadas, pero su funcionalidad completa no se implementa debido a lo extenso del alcance. Tampoco se establece la emulación del proceso de disociación debido a que el tema planteado aquí ya es considerablemente amplio. Tomando en cuenta, que ya se han analizado vulnerabilidades sobre el protocolo DAA en [7], no se hace necesario ejecutar el análisis sobre este protocolo pero si resulta imprescindible su implementación.

1.3. MARCO TEÓRICO

1.3.1. INTRODUCCIÓN A LAS WBAN

El cuerpo humano a pesar de ser una entidad que interactúa con su alrededor, es en cierto modo, un sujeto independiente y separado del resto, cuya complejidad requiere que se lo trate como tal. En este sentido, el cuerpo humano va más allá de una pequeña escala, es un ambiente que requiere que cubran sus necesidades para mantenerse vivo y confortable.

Con los constantes avances en el desarrollo de nuevas tecnologías que permiten miniaturización de dispositivos, bio-compatibilidad, transmisión inalámbrica de baja potencia entre otros, se ha dado paso al advenimiento de nuevas aplicaciones, cuyo enfoque está orientado a las necesidades del cuerpo humano y se apoyan en las Redes de Área Corporal o WBAN. Dichas redes consisten de nodos que pueden recolectar información del cuerpo humano y son del tipo implantables o portátiles [10]. Las WBAN se extienden a diferentes áreas de aplicación como pueden ser: atención médica, deportes y entretenimiento, sector militar-industrial y campo público-social [11].

El grupo de trabajo IEEE 802 estandarizó las WBAN a través del IEEE 802.15.6, con la finalidad de normalizar la comunicación dentro del área circundante del cuerpo humano, con un consumo de potencia optimizado y altamente confiable. La confiabilidad del estándar se determina a través de una estructura de seguridad que las entidades autorizadas deben establecer.

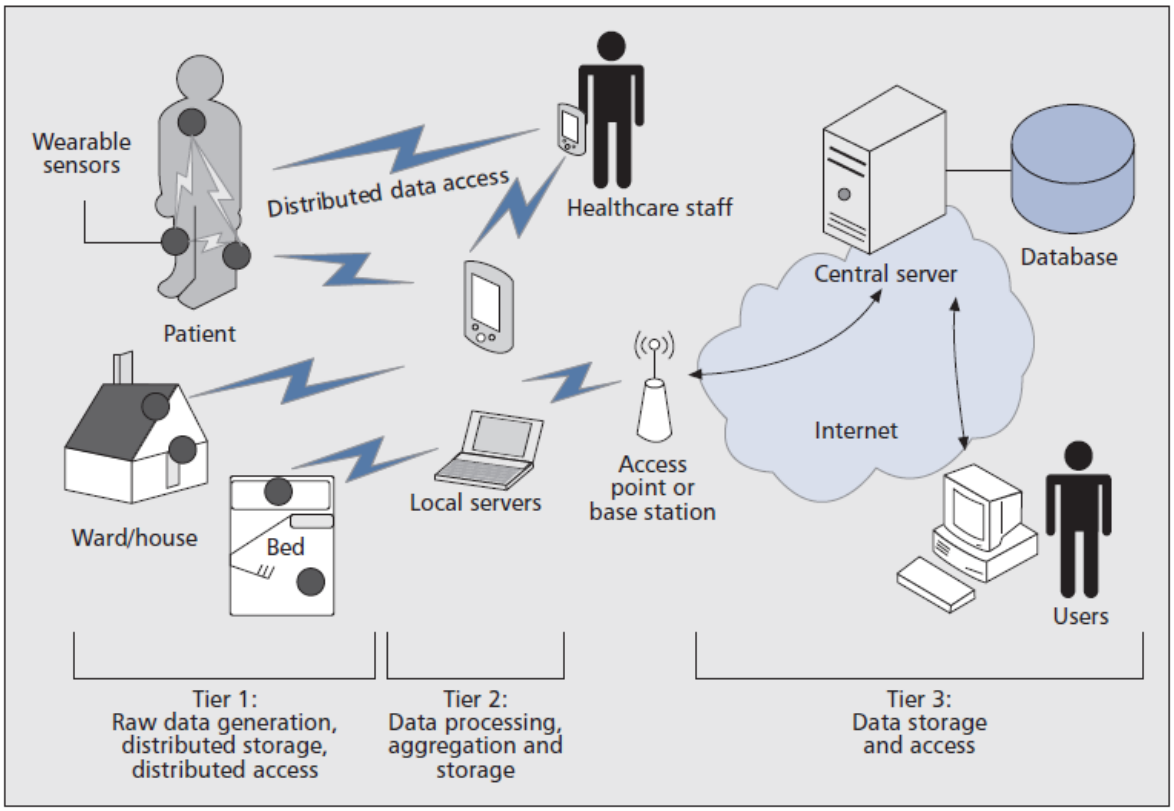


Figura 1.3. Representación de una arquitectura WBAN [10].

La Figura 1.3 muestra el concepto básico de la arquitectura WBAN, donde un individuo contiene diferentes sensores alrededor de su cuerpo, los cuales realizan una actividad de recopilación de información determinada. Después que los sensores han recopilado

información relacionada al usuario, se transmite a uno o más servidores locales. Por último, estos datos pueden ser transmitidos a bases de datos de salud centralizadas para registros permanentes.

Los nodos en el cuerpo permiten la obtención de información y su procesamiento para luego ser transmitida de forma inalámbrica a una LPU (*Local Processing Unit*). La LPU recopila y procesa los datos de todos los sensores para después transmitirlos a un servidor a través de una red WLAN, Bluetooth o telefonía móvil [12].

1.3.1.1. Aplicaciones de las WBAN

Como se mencionó anteriormente, existen diferentes áreas que requieren las características de las WBAN para su aplicación. Algunos artículos como [13], [14], dividen las aplicaciones en dos categorías, las cuales son: médicas y no médicas. En la Tabla 1.1 se indican algunas de las aplicaciones más relevantes de estas dos categorías, las cuales serán descritas a continuación.

Tabla 1.1. Aplicaciones WBAN

	CATEGORÍA	
	MÉDICA	**NO MÉDICA**
APLICACIONES	Telemedicina	Deportes
	Monitoreo remoto de pacientes	Militar
	Rehabilitación y terapia	Entretenimiento
	Bio-retroalimentación	Biometría
	Ambiente asistido	Gestión de notificaciones

Aplicaciones Médicas

La población humana consiste de diversos grupos que pueden padecer distintas enfermedades crónicas que necesitan un monitoreo constante de actividades físicas, fisiológicas etc. Es por ello que la inclusión de la tecnología WBAN en el cuidado de la salud es realmente atractiva, tanto para pacientes como médicos, cuyo resultado impactaría en una gran cantidad poblacional y mejoraría la atención médica para múltiples personas.

La obtención de los datos a través de los dispositivos dispuestos, permitiría una detección temprana y por lo tanto un diagnóstico y tratamiento más efectivo. Además, los datos adquiridos podrían almacenarse para ser analizados y retroalimentarse de ellos, construyendo nuevos sistemas de ayuda.

Adicionalmente con este sistema, una persona no necesita permanecer en cama para estar constantemente monitoreado, aumentando la movilidad del paciente y mejorando su calidad de vida.

En cuanto a seguridad, es importante que la información obtenida, sea tratada como sensible, debido a los daños que se pueden provocar si esta fuese vulnerada de alguna forma.

A continuación, se detallan las aplicaciones médicas de la Tabla 1.1.

- **Telemedicina**

Una contribución para el desarrollo de la atención médica a nivel mundial se encuentra en el perfeccionamiento de la telemedicina, la cual se enfoca en brindar servicios médicos a distancia mediante la utilización de tecnologías de la información y los sistemas integrados de salud. Los beneficios de la telemedicina incluyen un diagnóstico remoto, monitoreo de pacientes, apoyo en vivo por parte de médicos especialistas, permitiendo a los profesionales médicos atender a más pacientes.

- **Monitoreo remoto de pacientes**

Este tipo de aplicación se basa en monitorear los signos vitales del paciente que se encuentra en un hospital en condiciones críticas y cuyos parámetros son registrados para posteriormente ser utilizados por los profesionales de la salud. También se incluye el monitoreo de recuperación en el hogar, de las personas que han sido sometidas a algún procedimiento de riesgo y se requiere darle seguimiento para su óptima recuperación.

- **Rehabilitación y terapia**

La tecnología WBAN puede ser utilizada para empezar una rehabilitación después de haber sufrido algún tipo de accidente como por ejemplo una caída donde existiese rotura de algún hueso o ligamento. Un conjunto de dispositivos sensores puede encargarse de la tarea de rehabilitación y terapia posterior a la intervención quirúrgica, donde se proporcionen comentarios a través de los dispositivos de la rutina diaria que debe llevar el paciente.

- **Bio-retroalimentación**

Los usos de sensores para medir las actividades fisiológicas permiten al individuo controlar o cambiar sus actividades fisiológicas. La bio-retroalimentación admite controlar estados emocionales y funciones corporales como, por ejemplo: la migraña, estrés, mediante la personalización del ambiente para mejorar las condiciones del individuo con música,

iluminación, temperatura u otros factores que derivan de la información obtenida del usuario.

- **Ambiente asistido**

En este tipo de aplicación los sensores pueden ser del tipo *wearables*, los cuales permiten mejorar la calidad de vida de las personas tratando de mantener un estilo de vida independiente. Ayuda a los pacientes, quienes no requieren de una constante vigilancia, permanecer en casa, mientras el hospital les provee soporte y monitoreo. En caso de emergencia los sensores pueden levantar una alarma al centro médico más cercano.

Aplicaciones No Médicas

El escenario para aplicaciones no médicas también tiene un amplio campo de atención para las WBAN, las cuales igualmente requieren que la información que aquí se obtiene sea debidamente resguardada. Como se muestra en la Tabla 1.1 existen varias áreas donde se puede aplicar las WBAN, las cuales se detallan a continuación.

- **Deportes**

Se pueden utilizar dispositivos del tipo *wearables* para ser empelados en la monitorización del estado físico, el rendimiento y también el bienestar del deportista. Esto permitiría a los atletas mejorar su estatus, así como también sus destrezas.

- **Militar**

En el ámbito militar las WBAN también pueden ser utilizadas para medir el estado de salud de las personas, características fisiológicas entre otras utilidades; por ejemplo, se puede localizar la fuente de una explosión usando sensores de onda de choque, aceleración y el peso del soldado [13].

- **Entretenimiento**

Existen diferentes formas en las que las WBAN pueden proporcionar sus servicios en cuanto a entretenimiento se refiere. Por ejemplo, se puede implementar un sistema de música mediante dispositivos del tipo *wearables*. Otro ejemplo de entretenimiento, se tiene con SMARTeacher, cuyo software a más de entretenimiento también provee aprendizaje; este software utiliza un dispositivo como sensor de emociones. El juego utiliza señales de emociones, para controlar si el niño tiene frustración y así volverlo más fácil o si se aburre entonces se aumentará el nivel de dificultad [13].

- **Biometría**

La biometría se refiere a la autenticación de un individuo por medio de sus rasgos o conducta. La información biométrica recolectada de un individuo es única, la cual permite identificarlo y describirlo. La biometría basada en WBAN puede ser utilizada por ejemplo para la banca, acceso de información segura, desbloqueo de teléfonos inteligentes.

- **Gestión de notificaciones**

El concepto base de la gestión de notificaciones ocurre en el intercambio de información de persona a persona por medio de información contextual, como, por ejemplo, el lenguaje corporal, de tal forma que se puedan desarrollar herramientas de notificación minimizando la interrupción del usuario y maximizando la productividad.

1.3.1.2. Servicios de Seguridad para una WBAN

Con el creciente uso de la tecnología de comunicaciones en la medicina, también ha aumentado la preocupación sobre la privacidad de la información recopilada. A medida que la información es coleccionada por sistemas de comunicaciones, es necesario que se establezcan mecanismos adecuados para mantener segura la información.

Los datos recopilados de los pacientes son críticos en un tratamiento médico efectivo y cualquier modificación puede provocar desde un mal diagnóstico hasta la inestabilidad en la salud del usuario. Es por ello que, la seguridad y privacidad en una WBAN representa un tema importante y debe cumplir requisitos de seguridad para garantizar que la información esté protegida contra divulgaciones no autorizadas, así como modificaciones accidentales o intencionales, permitiendo que los recursos sean accedidos y utilizados solo por personas autorizadas.

A continuación, se describen los servicios de seguridad básicos de una WBAN [15], [16] .

- **Confidencialidad de los datos**

La recomendación ITU-T X.814 [17], establece que el objetivo de la confidencialidad es asegurar que la información se encuentre disponible solo para quienes tienen autorización.

La confidencialidad permite proteger la información ante divulgaciones ilegales, representando los datos de tal manera que sean legibles solo por las entidades autorizadas. Por ejemplo, en aplicaciones médicas cuando un atacante está escuchando la transmisión, puede llegar a obtener información relevante de un paciente violando de esta manera su derecho a la privacidad.

- **Autenticación de los datos**

De acuerdo a la ITU-T X.811 [18], el servicio de autenticación permite corroborar la identidad en el contexto del origen de los datos y de la entidad de quien los envía, bajo un escenario de comunicación.

El objetivo de la autenticación en una WBAN, es asegurar que la fuente, que es parte del proceso de comunicación, es quien dice ser. Dado que un atacante puede enviar paquetes falsos, es necesario que se realice la verificación del origen de los datos.

- **Integridad de los Datos**

El objetivo de la integridad de acuerdo a la ITU-T X.815 [19], es proteger los datos contra: modificación, supresión, creación, inserción o reproducción no autorizada.

Dado que la información que se transmite en una WBAN es a través de un canal de comunicación inseguro, la información de un usuario puede ser alterada por un atacante. Por lo tanto, la integridad de los datos es imprescindible para evitar que una persona no autorizada modifique la información; la falta de este servicio puede provocar severos daños cuando, por ejemplo, se trata de un diagnóstico médico o del suministro de atención al paciente.

- **Data Freshness**

El *data freshness* o defensa ante la reproducción, asegura que los datos recibidos no sean réplicas de datos antiguos, es decir que un receptor solo procese datos recientes. Siendo así que, si un atacante envía datos que ya han sido procesados anteriormente, los dispositivos que reciban esta información puedan detectar esta anomalía y protegerse descartando esta información replicada. El mantener el orden del marco de los datos previene la reutilización de los mismos.

- **Disponibilidad**

La disponibilidad de los datos, se refiere a que los datos deben estar disponibles todo el tiempo, considerando que el monitoreo es en tiempo real. No hay que pensar que la disponibilidad depende de un servicio de seguridad en específico; por lo tanto, para mantener la disponibilidad es necesario que los servicios de seguridad mencionados anteriormente formen un conjunto coherente que permitan la disponibilidad.

1.3.1.3. Mecanismos de Seguridad en una WBAN

Los servicios y mecanismos de seguridad permiten proteger a las redes de ataques tales como la escucha clandestina, negación de servicios, falsificación, corrupción de información, acceso no autorizado, suplantación [20].

Para implementar un servicio de seguridad es necesario utilizar uno o más mecanismos. A continuación, se describen algunos mecanismos de seguridad acorde a los servicios de seguridad descritos para una WBAN.

- **Cifrado**

Permite proveer de confidencialidad a la información y además sirve como complemento de otros mecanismos de seguridad [21].

Las técnicas criptográficas son utilizadas en los distintos servicios de seguridad para mantener la confidencialidad, integridad, autenticación, etc. de los datos, mediante el cifrado, el cual consiste en la transformación de texto en una secuencia inteligible y que solo puede ser entendida una vez que ha sido descifrada.

La criptografía puede ayudar en la protección contra [21]:

- La modificación u observación de mensajes.
- El análisis del tráfico.
- El repudio[2].
- La falsificación.
- La conexión no autorizada.

- **Gestión de claves**

Por lo general, cuando se utilizan mecanismos de cifrado se requieren mecanismos de gestión de claves. La gestión de claves se encarga de [21]:

- Una generación adecuada de claves, dependiendo del nivel de seguridad que se necesite.
- El establecimiento de las entidades autorizadas que deben recibir una copia de cada clave.

[2] Se puede evitar el repudio de un mensaje con criptografía, ya que el emisor es el único con la clave y no puede reclamar después que no fue quien envió el mensaje.

- El reparto seguro de las claves a las entidades autorizadas.

Otros de los puntos que se deben tener en consideración respecto a la gestión de claves son: el tiempo de vida de la clave, el tipo de función a la cual se encuentra reservada y la distribución de la clave. La distribución de las claves también puede realizarse de manera física, tomando las consideraciones de seguridad necesarias para realizarlo.

- **Firma Digital**

La firma digital es un mecanismo que utiliza algoritmos criptográficos asimétricos, que permiten al receptor del mensaje autentificar quien originó el mensaje y verificar que este no ha sido modificado desde el momento que se produjo. La firma digital ayuda en la integridad de los datos y autenticación del origen.

Este mecanismo conlleva dos procedimientos: el cifrado y la verificación de los datos. El primero utiliza información privada del firmante como clave privada y el segundo utiliza procedimientos e información pública para verificar la firma [21].

- **Número de Secuencia**

El número de secuencia es un mecanismo que utiliza una política para numerar mensajes, donde las partes involucradas usan esta política para detectar la reproducción de los mensajes; el objetivo es que un mensaje con un número particular se acepte solo una vez dentro de un periodo establecido [22]. Este mecanismo ayuda en general a la integridad de los datos, pero de manera particular al servicio de *data freshness*.

- ***Nonce***

El *Nonce* es un mecanismo que utiliza valores únicos no necesariamente aleatorios por cada mensaje enviado. Por lo general lleva una secuencia de incremento de valores ordenados. Dichos valores son utilizados en el servicio de seguridad *data freshness*.

- **MAC (Message Authentication Code)**

MAC o Código de Autenticación de Mensajes es una etiqueta que provee de integridad y autenticidad a los mensajes que se transmiten de un emisor a un receptor. Esta etiqueta va junto al mensaje que se desea transmitir. Para generar una MAC es necesario del mensaje a ser transmitido combinado con una clave secreta. Un destinatario puede generar la misma MAC si posee la clave secreta, ya que un mensaje dado siempre producirá la misma MAC [23].

La Tabla 1.2 muestra la colaboración que dan los mecanismos de seguridad para que se cumplan los servicios de seguridad que se han mencionado en esta sección.

Tabla 1.2 Correspondencia de los Servicios y Mecanismos de Seguridad

SERVICIO	MECANISMO					
	Cifrado	Gestión de claves	Firma digital	Número de Secuencia	*Nonce*	Message Autentication Code
Confidencialidad	X	X				
Autenticación	X		X			X
Integridad	X			X		
Data Freshness	X			X	X	
Disponibilidad	X	X	X	X	X	X

1.3.2. ESTÁNDAR 802.15.6

El estándar IEEE 802.15.6 es una norma de comunicación inalámbrica en las proximidades o dentro del cuerpo humano, pero no limitada a los humanos [2]. Las bandas ISM (*Industrial, Scientific, Medical*) y otras son utilizadas por el estándar.

El estándar define un rango para la tasa de bits de entre 10 Kbps y 10 Mbps, dependiendo de la aplicación y ubicación de los nodos. Además, también establece criterios estrictos para QoS (*Quality of Service*), potencias extremadamente bajas y considera los efectos de la transmisión y recepción de información a causa de la presencia de una persona y los cambios que pueden sufrir todas estas características a causa del movimiento propio de las personas. Los principales requerimientos para una WBAN según el estándar se muestran en la Tabla 1.3 [7], [24].

Tabla 1.3 Requerimientos de una WBAN según el estándar IEEE 802.15.6 [2]

REQUERIMIENTO	DESCRIPCIÓN
PER (*Packet Error Rate*)	≤ al 10% con un *Payload* de 255 octetos.
Asociación/Disociación de nodos	Tiempo menor a 3 segundos
Número de nodos en una WBAN	256 nodos
Jitter	Menor a 50 ms
Latencia	Menor a 125 ms en aplicaciones médicas y menor a 250 ms para el resto.
Número de redes soportadas en la capa física	Hasta 10 en un volumen de 6 m^3
Potencia de transmisión de los nodos	0.1 mW
Potencia máxima radiada	1 mW

Además de los requerimientos mencionados en la Tabla 1.3, es importante tener en consideración otras características que deben cumplir las WBAN como: la confiabilidad en

la comunicación cuando existe movimiento, mecanismos de ahorro de energía, coexistencia de los nodos dentro o sobre el cuerpo humano, convivencia con redes de otras tecnologías, entre otras.

1.3.2.1. Consideraciones Generales

Dentro de este ámbito se identifican las bases de funcionamiento del estándar IEEE 802.15.6, se considerarán temas como topología de red y las capas presentes en el estándar.

- **Topología de Red**

El estándar define sus componentes de red como nodos y *Hub*s, los cuales pueden estar organizados en conjuntos lógicos denominados BAN, en donde el *Hub* controla el acceso al medio de los nodos en su red y suministra energía en la misma. Existe solo un *Hub* por BAN.

La topología de red que se establece es de tipo estrella y puede ser de un solo salto o de dos; en la de un solo salto la trama se intercambia entre el *Hub* y el Nodo, mientras que, en la segunda, el *Hub* y Nodo intercambian información a través de un nodo de retransmisión.

La Figura 1.4 muestra los dos tipos de topologías estrella conectadas a través de un mecanismo que permite la convivencia de las redes. La WBAN 1, es la topología de red estrella de un solo salto; mientras que la WBAN 2, corresponde a la topología estrella de dos saltos que incluye dos nodos de retransmisión. Cada una de las redes tiene su correspondiente nodo coordinador, *Hub*1 y *Hub*2.

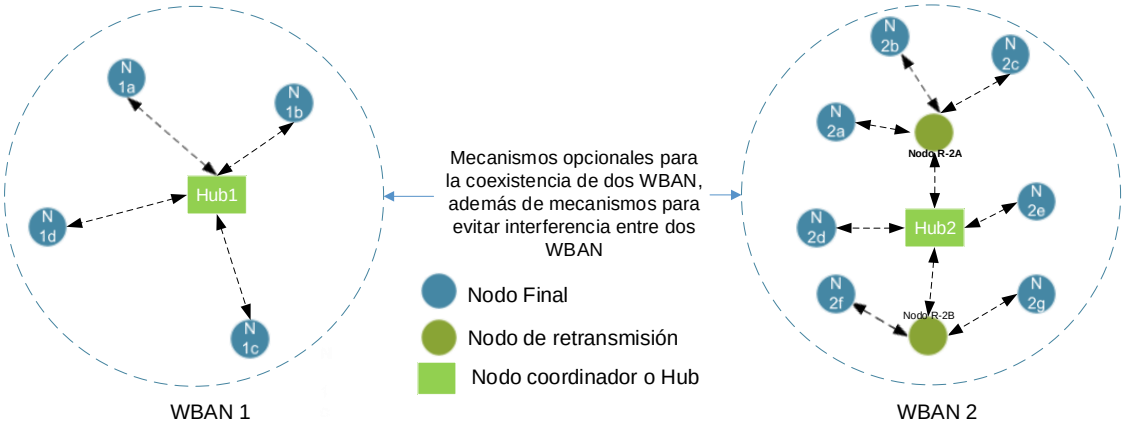


Figura 1.4. Topologías de red del estándar IEEE 802.15.6 [2]

Es importante mencionar que las topologías de dos saltos requieren menor consumo de energía, debido a que la potencia de transmisión es menor para alcanzar el nodo coordinador, y se tiene menor interferencia por la cercanía de los nodos. Pero existen otras características negativas en comparación con las de un solo salto, siendo una de ellas el aumento del retardo para los nodos que deben transmitir a través de un nodo retransmisor y si un nodo retransmisor falla, entonces los otros nodos conectados a él también fallarán.

- **Capas**

El estándar IEEE 802.15.6 define una subcapa MAC (*Medium Access Control*) para tres tipos de capa PHY *(Physical layer*), las cuales son [25]: NB (*Narrowband*), UWB (*Ultra-wideband*) y HBC (*Human Body Communications*). Tanto la capa PHY como la subcapa MAC permiten una comunicación con bajo consumo energético, corto alcance y alta complejidad. Las capas PHY mencionadas, se diferencian por las bandas de frecuencia utilizadas en las distintas aplicaciones para WBAN.

Dentro de las responsabilidades de la capa PHY en el estándar se encuentran: la activación y desactivación del radio *transceiver*[3], evaluación del canal, recepción y transmisión, corrección de errores y modulación de datos [26].

La NB-PHY está orientada para operaciones BAN sobre (*on-body)* y operaciones BAN dentro (*in-body)* como en los implantes. Enfocada principalmente en aplicaciones médicas de corto alcance con enlaces altamente confiables, donde al menos un extremo está sobre o dentro del cuerpo [27].

UWB-PHY cubre solo operaciones BAN *on body*. Reduce la complejidad de implementación, lo cual apoya para establecer un bajo consumo de energía. Provee niveles de potencia segura acorde a la banda MICS (*Medical Implant Communications Service*). UWB puede ser utilizada en aplicaciones médicas como no médicas [27].

Mientras tanto HBC, al igual que UWB, cubre solo operaciones BAN *on-body*. Una característica importante en HBC es el enlace de comunicación empleado, en el cual las ondas electromagnéticas utilizan la superficie del cuerpo humano como medio de propagación; se usan electrodos para este propósito [27].

La subcapa MAC se encarga de la operación de control de todo el sistema como la coexistencia de redes, seguridad de la BAN, comunicación entre los dispositivos de una BAN, QoS, etc. Como se mencionó anteriormente en esta sección, los nodos se encuentran

[3] *Transceiver* o también conocido como transceptor es un dispositivo que transmite y recibir señales que pueden ser de radio o eléctricas.

organizados por un nodo coordinador o *Hub*, el cual se encarga de disponer el acceso al medio mediante uno de los siguientes modos:

- Modo *Beacon* con límites de supertrama.
- Modo *No-Beacon* con límites de supertrama.
- Modo *No-Beacon* sin límites de supertrama.

En el modo *Beacon*, el coordinador transmite tramas *Beacon* para establecer los límites de la supertrama y así, permitir el control de asociación a la red y sincronismo de los nodos; mientras que en el modo *No-Beacon* no se transmiten tramas *Beacon*, pero es necesario un tiempo de referencia por lo cual el *Hub* envía tramas de poleo para comunicar tales límites a los nodos. El mecanismo de acceso para los nodos puede ser programado en el caso que se utilice supertrama y no programado o CSMA/CA () cuando se utiliza el modo sin supertrama.

1.3.3. ESTRUCTURA DE SEGURIDAD DEL ESTÁNDAR IEEE 802.15.6

1.3.3.1. Niveles de seguridad

El estándar define tres niveles de seguridad para que las entidades autorizadas (Nodos y *Hub*) puedan transmitir tramas de datos en la BAN. Estos niveles de seguridad son:

- Nivel 0 – Comunicación Insegura

Es el nivel más bajo y por lo tanto no proporciona ningún servicio de seguridad. Aquí la información se transmite en tramas inseguras.

- Nivel 1 – Autenticación sin Cifrado

Este nivel ya proporciona algunos servicios de seguridad para la información que es transmitida como la autenticación, defensa ante reproducción de mensajes y validación de la integridad. No obstante, los servicios como la confidencialidad y protección de la privacidad están ausentes en este nivel. Los datos son transmitidos en tramas autenticadas, pero no cifradas.

- Nivel 2 – Autenticación y Cifrado

Este nivel de seguridad es el más alto, en donde los datos se transmiten en tramas seguras con autenticación y que además se encuentran cifradas. El nivel 2 complementa los servicios de seguridad del nivel 1, es decir incluye confidencialidad y protección de la privacidad.

Nodos y *Hub*s en una comunicación deben realizar una transición a través de ciertos estados como se muestra en la Figura 1.5, antes de poder intercambiar información de usuario. Los estados en la Figura 1.5 indican las tramas que se deben intercambiar entre nodo y *Hub*.

En caso de éxito en la transmisión de tramas, se indica el estado al cual debe avanzar o al que debe retroceder si no se ha superado el proceso.

Cuando la comunicación se instaura como segura en el nivel más alto, se establecen cuatro estados por los cuales, tanto Nodo como *Hub* deben atravesar. Cuando se encuentran en el estado *Orphan* no existe relación entre los componentes de la red, y solo se permite intercambio de tramas de Asociación de Seguridad para establecer una asociación segura, que se refiere a la generación de una clave maestra MK (*Master Key*). En caso de que el proceso falle al generar una MK, no se permite avanzar al siguiente estado *Associated.*

Cuando Nodo y *Hub* avanzan al estado *Associated* solo se permite transmisión de tramas no seguras PTK, Disociación de Seguridad y tramas de control no seguras.

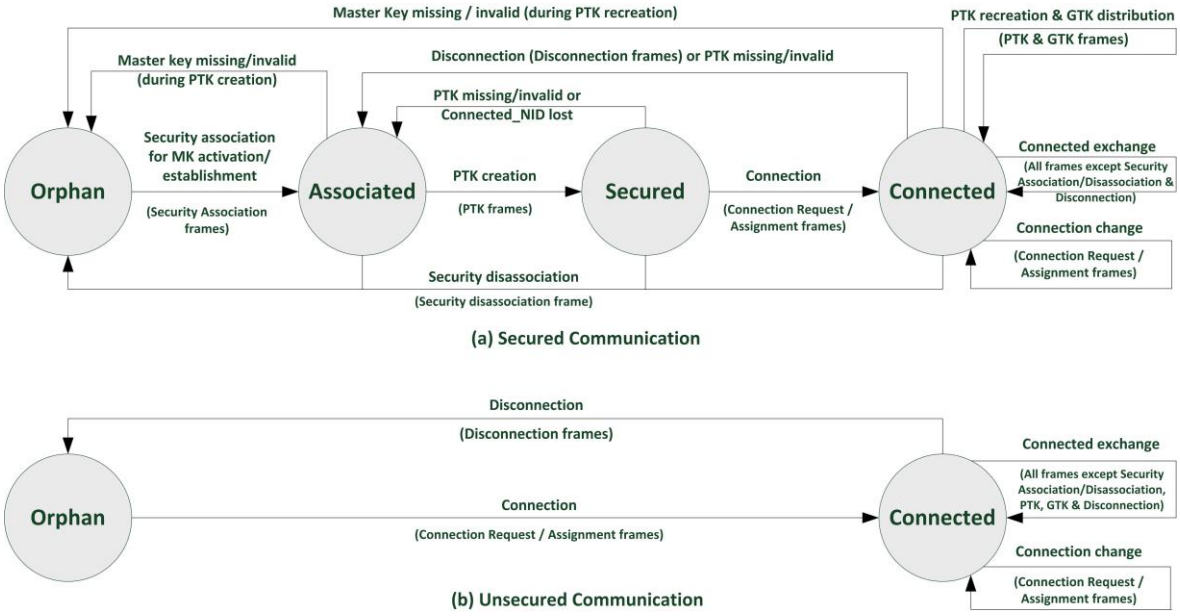


Figura 1.5. Diagrama de estados de seguridad [2].

Para pasar al estado *Secured*, se debe completar con éxito el proceso de Creación PTK, el cual establece la creación de un PTK con el intercambio de tramas inseguras PTK. Se retorna al estado *Orphan* cuando se recibe una trama de Disociación de Seguridad o por problemas de generación de la MK o PTK. En el estado *Secured* tanto el Nodo como el *Hub* poseen un PTK para asegurar el intercambio de tramas; en este estado solo se permite

el intercambio de tramas de Solicitud de Conexión, Asignación de Conexión, Disociación y tramas de control seguras, menos las de Asociación. Si las tramas de conexión han sido transmitidas con éxito se pasa al estado *Connected,* caso contrario se regresa al estado anterior dependiendo de las fallas ocurridas en el proceso de Creación PTK. En el estado *Connected,* tanto el Nodo como el *Hub* pueden transmitir tramas seguras excepto las de Asociación de Seguridad y solo se permiten tramas no seguras de control cuando no se ha establecido autenticación de tramas tipo control durante la asociación.

En una comunicación insegura se omiten todos los procesos intermedios entre el estado *Orphan* y el estado *Connected* como se puede ver en la Figura 1.5. Para pasar de un estado al otro se requieren tramas de Solicitud de Conexión y Asignación de Conexión. También se permite intercambio de tramas de control inseguras. Para poder regresar al estado *Orphan* es necesario utilizar una trama de Desconexión.

1.3.3.2. Paradigma de seguridad

Una comunicación segura puede realizarse solo entre un par de componentes, Nodo y un *Hub*, la cual será del tipo *unicast*; mientras que, cuando esta se realiza de un *Hub* a varios Nodos, será del tipo *multicast*. La comunicación segura *multicast* requiere que primero se establezca la comunicación *unicast*. Para dicho establecimiento, tanto el Nodo como el *Hub* deben seguir el paraddigma de seguridad que se muestra en la Figura 1.6.

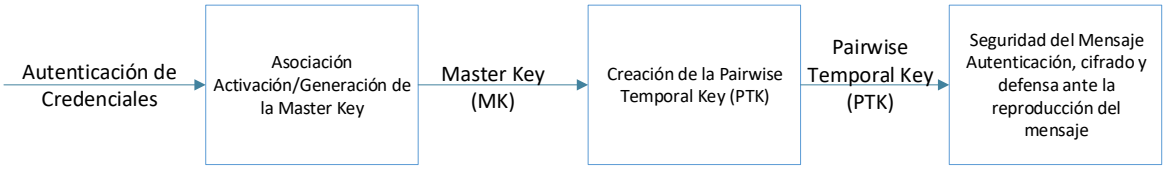


Figura 1.6. Paradigma de Seguridad del estándar IEEE 802.15.6 [2].

- **Asociación**

El estándar define cuatro protocolos para el proceso de Asociación para una comunicación segura, de los cuales solo se escoge uno para llevar a cabo el paradigma de seguridad. Dentro de los protocolos de Asociación establecidos se tienen: *Unauthenticated Association, Public Key Hidden Association, Password Authenticated Association, Display Authenticated Association*. Todos los protocolos de este proceso tienen como objetivo el establecimiento de una MK entre las entidades Nodo y *Hub*.

En este proyecto de titulación, para cumplir el paradigma de seguridad se utilizará el protocolo *Display Authenticated Association.*

El proceso de Asociación se encarga de la generación de una MK, para lo cual el protocolo define algunos mecanismos para llevar a cabo esta tarea de manera segura. Además, este protocolo establece que tanto el Nodo como el *Hub* requieren tener un *display* para mostrar un número decimal, y antes de aceptar una MK, el usuario debe verificar que el número mostrado en ambas partes sea el mismo.

Es posible dividir al protocolo en tres etapas para una mejor comprensión del mismo, las cuales son [7]:

- Fase de creación de elementos de asociación y proceso Diffie Hellman.
- Fase de creación de elementos de autenticación con los elementos del proceso Diffie-Hellman.
- Fase de autenticación de entidades.

En la primera fase se crean elementos tales como: una llave secreta de 256 bits (SK) establecida partir de una función randómica, una llave pública (PK) que se obtiene a partir de curva criptográfica elíptica, un *Nonce*[4] de 128 bits también aleatorio, un *Witness*[5] creado por el Nodo con CMAC.

En la segunda fase se calculan algunos elementos y se ejecutan procedimientos. En primera instancia se debe calcular la llave Diffie Hellman necesaria para la autenticación; una vez que se obtiene este elemento, se obtienen los 128 bits más significativos de esta llave y a este nuevo conjunto de bits se lo denominará Temp, el cual, junto con el *Witness* de la etapa anterior y otros elementos servirá para calcular con CMAC el MK_KMAC (*Key Message Authentication Code*) de las correspondientes entidades.

En la tercera fase se calculan algunos elementos y se ejecutan procedimientos. En primera instancia se comparan los MK_KMAC de las entidades y si son iguales se pasa a comparar el *Witness* de las entidades; si son iguales se creará el *display* que tiene que ser verificado por el usuario, tanto en el Nodo como en el *Hub*. Si esto es así, entonces se crea el MK en cada entidad.

- **Creación PTK**

Para que empiece el proceso de Creación PTK, se establece como condición que se haya generado y activado una MK de 128 bits en el proceso de Asociación.

[4] ***Nonce***: dentro del proceso de Asociación es un valor aleatorio de 128 bits.
[5] ***Witness:*** elemento utilizado para la autenticación de entidades.

A este proceso también se puede dividir en fases, las cuales se mencionan a continuación:

- Fase de creación.
- Fase de autenticación.

En la fase de creación se generarán y crearán los elementos que se muestran en la Figura 1.7.

***Nonce*:** es un número escogido al azar, de 128 bits que se crea en cada entidad (Nodo-*Hub*).

Intercambio de la primera y segunda trama: Para que se puedan crear los siguientes elementos es necesario que se comparta el *Nonce* tanto del Nodo como del *Hub* a través de la primera y segunda trama PTK según corresponda.

Creación PTK: Obtención de la llave segura a partir de la Ecuación 1.1, que será activada en la siguiente fase.

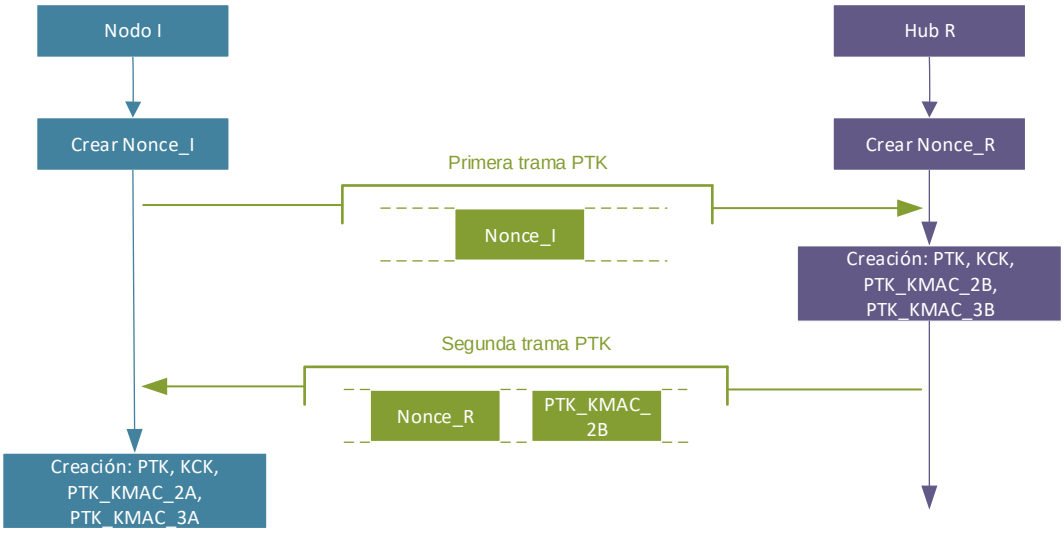


Figura 1.7. Fase de creación.

$$PTK = CMAC\ ($$

$$MK, Direccion\_I\ ||\ Direccion\_R\ ||\ Nonce\_I\ ||\ Nonce\_R\ ||\ PTK\_Control, 128) \qquad (1.1)$$

Creación del KCK (*Key Confirmation Key*): Valor de 128 bits que sirve como llave para asegurar el PTK_KMAC. Se calcula a partir de la Ecuación 1.2.

$$KCK = CMAC\ ($$

$$MK, Direccion\_R\ ||\ Direccion\_I\ ||\ Nonce\_R\ ||\ Nonce\_I\ ||\ PTK\_Control, 128) \quad (1.2)$$

Creación del PTK_KMAC_2B: Corresponde a los 64 bits más a la izquierda del valor obtenido en la Ecuación 1.3 y sirve como elemento de autenticación. Se muestra en la Ecuación 1.4.

$$P = CMAC\ ($$

$$KCK, Direccion\_I\ ||\ Direccion\_R\ ||\ Nonce\_R\ ||\ Nonce\_I\ ||\ PTK\_Control, 128) \quad (1.3)$$

$$PTK\_KMAC\_2B = LMB\_64\ (P) \quad (1.4)$$

Creación del PTK_KMAC_3B: Corresponde a los 64 bits más a la derecha del valor obtenido en la Ecuación 1.3 y sirve como elemento de autenticación. Se muestra en la Ecuación 1.5.

$$PTK\_KMAC\_3B = RMB\_64\ (P) \quad (1.5)$$

Creación del PTK_KMAC_2A: Corresponde a los 64 bits más a la izquierda del valor obtenido en la Ecuación 1.3 y sirve como elemento de autenticación. Se muestra en la Ecuación 1.6.

$$PTK\_KMAC\_2A = LMB\_64\ (P) \quad (1.6)$$

Creación del PTK_KMAC_3A: Corresponde a los 64 bits más a la derecha del valor obtenido en la Ecuación 1.3 y sirve como elemento de autenticación. Se muestra en la Ecuación 1.7.

$$PTK\_KMAC\_3A = RMB\_64\ (P) \quad (1.7)$$

En la segunda fase de autenticación, se verifica si los elementos de seguridad creados son iguales. La Figura 1.8 muestra el desarrollo de esta fase.

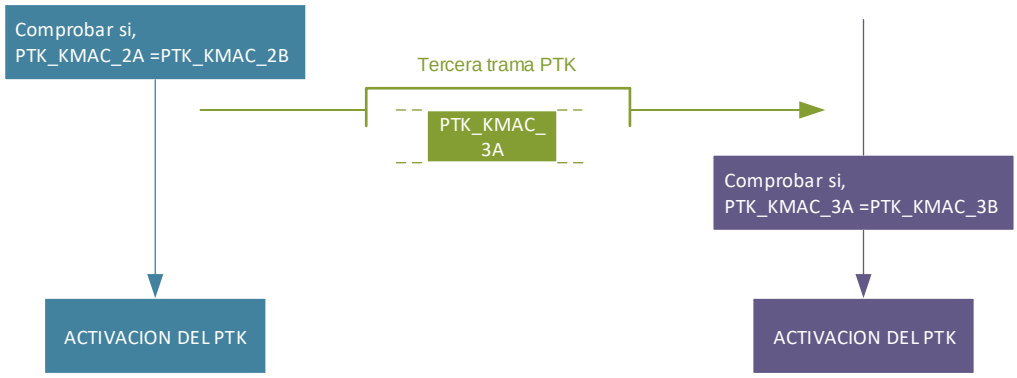


Figura 1.8. Fase de autenticación y activación del PTK.

Comprobación PTK_KMAC_2: Si PTK_KMAC_2A y PTK_KMAC_2B son iguales, se envía la tercera trama con el PTK_KMAC_3A del Nodo.

Comprobación PTK_KMAC_3: Si PTK_KMAC_3A y PTK_KMAC_3B son iguales, se activa el PTK tanto en el Nodo como en el *Hub*

- **Seguridad del Mensaje**

La seguridad del mensaje es el último proceso a cumplirse en el paradigma de seguridad y éste dependerá del nivel escogido en el proceso de Asociación. Para el presente trabajo de titulación se ha escogido el nivel más alto que corresponde al dos. A este proceso se lo puede dividir en fases que deben ser cumplidas para asegurar los datos transmitidos entre Nodo y *Hub*. Las fases que deberán cumplirse en la Seguridad del Mensaje son:

- Formación de elementos.
- Fase de autenticación.
- Fase de cifrado.
- Fase de descifrado.

En la fase de formación de elementos, se establecerán los elementos necesarios para cumplir los objetivos de las siguientes fases. La Figura 1.9 muestra los elementos que deberán formarse en cada entidad.

***Nonce*:** Elemento de 13 octetos requerido para la autenticación y cifrado. Está formado como muestra la Figura 1.10. El campo Número de Secuencia de Bajo Orden permite la actualización del mensaje y la detección del *replay*. Mientras que el campo Número de

Secuencia de Alto Orden sirve para evitar repetir los *Nonce*s, permitiendo asegurar un mayor número de mensajes.

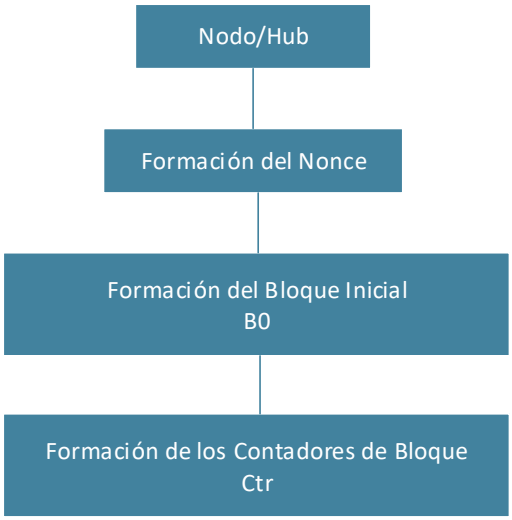


Figura 1.9. Formación de elementos.

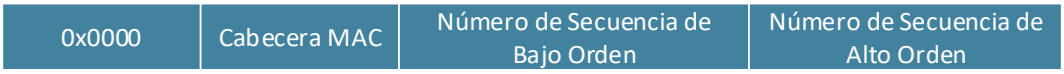


Figura 1.10. *Nonce* [2].

Bloque inicial B0: Elemento de 16 octetos requerido para la autenticación. Se compone del *Nonce* definido anteriormente y un campo que contiene la longitud del *Payload* de la trama de datos. La Figura 1.11 muestra sus campos.

Figura 1.11. Bloque inicial B0 [2].

Bloque contador Ctr: Elemento de 16 octetos requerido para autenticación y cifrado. Se compone del *Nonce* definido anteriormente y un contador que permite determinar el número de bloques de 16 octetos del *Payload*. La Figura 1.12 muestra sus campos.

Figura 1.12 Bloque contador Ctr [2].

Formación de Bloques B: Se obtiene a partir de la división en bloques de 16 octetos al texto plano, sí el último bloque no es múltiplo de 128 bits, se lo rellena con ceros para cuando se va a calcular el MIC (*Message Integrity Code*), no se rellena cuando se van a cifrar. Son utilizados tanto para la autenticación y cifrado.

En la fase de autenticación, se calcula la etiqueta MIC, utilizada para la autenticación del mensaje. Se basa en las Ecuaciones 1.8 y 1.9.

$$X_0 = AES(B_0), X_i = AES(B_i \oplus X_{i-1}), i = 1, \dots m \tag{1.8}$$

$$MIC = LMB\_n(M), M = AES(Ctr_0) \oplus X_m \tag{1.9}$$

La Figura 1.13 muestra el proceso de aplicar las Ecuaciones 1.8 y 1.9 para el cálculo de la etiqueta MIC.

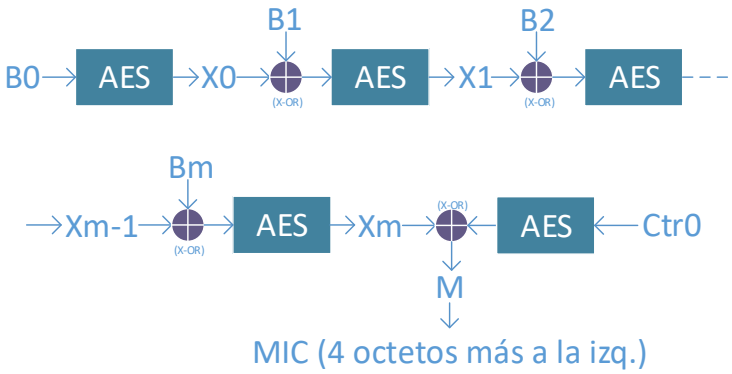


Figura 1.13. Proceso para obtener la etiqueta MIC [2].

En la fase de cifrado, los bloques B que contienen el texto plano, junto con los bloques Ctr, pasan por un proceso para ser encriptados como se muestra en la Figura 1.14. La Ecuación 1.10 se aplica para los bloques completos, mientras que la 1.11 para cifrar el último bloque en caso de que este sea incompleto. Por último, la Ecuación 1.12 une los diferentes bloques cifrados para obtener el *Payload*.

$$B'_i = B_i \oplus AES(Ctr_i), \qquad i = 1, \dots m-1 \tag{1.10}$$

$$B'_m = L\_n(B_m) \oplus L\_n(AES(Ctr_m)) \quad (1.11)$$

$$PayloadCifrado = B'_1 || B'_2 || \dots || B'_{m-1} || B'_m \quad (1.12)$$

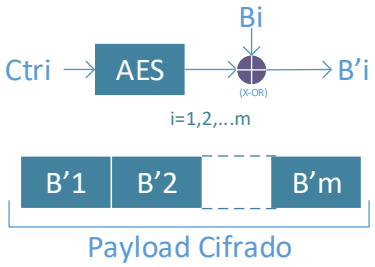


Figura 1.14. Proceso para obtener el *Payload* cifrado [2].

En la fase de descifrado, se realizan los procesos inversos de las fases de autenticación y cifrado descritos anteriormente. Para poder autenticar la trama, primero se realiza el descifrado del *Payload*, mediante la división en bloques de 16 octetos. Este proceso se realiza de acuerdo a las Ecuaciones 1.13 y 1.14 y 1.15 y se muestra su desarrollo en la Figura 1.15. La Ecuación 1.13 indica el descifrado de los bloques B' múltiplos de 128 bits, mientras que, para el último bloque, si este no es múltiplo de 128 bits se utiliza la Ecuación 1.14, finalmente, la Ecuación 1.15 indica la formación del *Payload* descifrado.

$$B_i = B'_i \oplus AES(Ctr_i), \qquad i = 1, \dots m-1 \quad (1.13)$$

$$B_m = B'_m \oplus L\_n(AES(Ctr_m)) \quad (1.14)$$

$$PayloadDescifrado = B_1 || B_2 || \dots || B_{m-1} || B_m \quad (1.15)$$

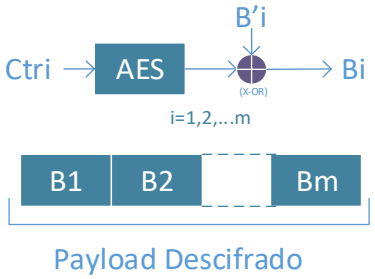


Figura 1.15. Proceso para obtener el *Payload* descifrado [2].

Si el MIC calculado por el *Hub* coincide con el enviado en la trama entonces la trama se autentica y se pueden considerar como verdaderos los bloques descifrados; pero si, por el contrario, los valores de etiquetas no concuerdan entonces la trama se considera falsa y se descarta.

1.3.4. MECANISMOS DE SEGURIDAD DEL PARADIGMA

1.3.4.1. Llaves públicas

El algoritmo Diffie Hellman utiliza criptografía de llave pública. Este algoritmo permite acordar una clave secreta entre dos entidades que utilizan un canal inseguro de comunicación. A este tipo de criptografía se la conoce como cifrado asimétrico porque utiliza dos tipos de llaves: llaves públicas y llaves privadas.

El par de entidades A y B, deben generar sus llaves públicas y privadas. Cuando se requiere intercambiar un mensaje, A y B deberán compartir sus llaves públicas, al ser públicas cualquiera puede conocer su valor. Para cifrar un mensaje que será enviado desde la entidad A, ésta utiliza la llave pública de B. El mensaje solo podrá ser descifrado con la llave privada de B. La llave privada es una clave secreta que solo la conoce la entidad que la generó.

Entonces, la criptografía asimétrica se basa en la creación de una llave pública por entidad, que será generada a partir de una llave privada que solo la conoce la entidad que creó esa llave. Para generar la llave pública se utilizan funciones matemáticas unidireccionales, evitando así que se pueda reproducir la llave privada a partir de conocer la llave pública.

1.3.4.2. Criptografía de curva elíptica

Es utilizado por sistemas criptográficos, gracias a que la cantidad de bits necesarios es menor que otros sistemas y la carga computacional para el cálculo de llaves también es menor. Es útil en ambientes donde la potencia de consumo, y procesamiento son limitadas.

La curva elíptica se define en base a la Ecuación 1.16. Para que la curva obtenga tres soluciones distintas, se debe controlar que la curva sea válida mediante la Ecuación 1.17.

$$y^2 = x^3 + ax + b \tag{1.16}$$

$$4a^3 + 27b^2 \neq 0 \tag{1.17}$$

En criptografía se utilizan curvas elípticas definidas sobre campos finitos Z_p, esto quiere decir que las curvas tienen un número finitos de puntos y cuyas coordenadas son números enteros. El tener números enteros como coordenadas, permite realizar los cálculos de manera eficiente y se evitan los errores que produce el redondeo si las curvas no estuviesen definidas sobre campos finitos. Al ser Z_p un conjunto finito entonces *p* será el un número primo mayor a 3. La curva elíptica definida sobre Z_p se define en la Ecuación 1.18. El conjunto finito Z_p se define en la Ecuación 1.19.

$$y^2 = x^3 + ax + b \ (mod\ p) \tag{1.18}$$

$$Zp = \{0,1,2, \dots \dots, p-1\} \tag{1.19}$$

Al igual que en curvas elípticas definidas sobre los números reales, para el campo finito Z_p, la curva elíptica debe cumplir con el discriminante, el cual se define en la Ecuación 1.20.

$$4a^3 + 27b^2 \neq 0 \ (mod\ p) \tag{1.20}$$

No todos los valores en Z_p pueden ser soluciones de la curva elíptica. Por lo tanto, la llave pública P_k se obtiene de acuerdo a la Ecuación 1.21.

$$S_k \times G = P_k \tag{1.21}$$

Donde:

S_k *:* Llave privada.

G: Punto solución de la curva elíptica.

En el estándar IEEE 802.15.6 la clave privada tiene 256 bits y además especifica los parámetros para la Curva P-256 que se encuentran estandarizadas en el FIPS Pub 186-3.

- $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$
- r=115792089210356248762697446949407573529996955224135760342422250610685120443 69
- $a = p-3$

- $b = 5ac635d8aa3a93e7b3ebbd55769886bc651d06b0cc53b0f63bce3c3e27d2604b$
- $Gx = 6b17d1f2e12c4247f8bce6e563a440f277037d812deb33a0f4a13945d898c296$
- $Gy = 4fe342e2fe1a7f9b8ee7eb4a7c0f9e162bce33576b315ececbb6406837bf51f5$
- $1 < Sk < r - 1$

Donde:

p: corresponde a número primo parte del campo finito.

r: orden del punto base G.

a, b: elementos del campo finito, definen la ecuación de la curva elíptica.

Gx, Gy: definen las coordenadas iniciales de la curva.

1.3.4.3. Estándar de Encriptación Avanzada (AES)

AES es un algoritmo criptográfico utilizado para datos que requieren seguridad, convirtiéndolos en datos no legibles; se asegura que ninguna persona no autorizada pueda develarlos. Este algoritmo utiliza cifrado de bloque simétrico para cifrar y descifrar información.

AES está en la capacidad de cifrar y descifrar datos de un tamaño de bloque igual a 128 bits, utilizando una de las siguientes llaves: 128, 192 o 256 bits [28].

El bloque de 128 bits a ser cifrado se lo puede ver como una matriz de estado de 4x4 bytes y que se encuentra organizado como se muestra en la Figura 1.16.

Figura 1.16. Matriz de estado AES [29].

Para una clave de 128 bits se deben generar 10 subclaves a partir de la clave inicial K_0. Cada una de las subclaves (incluida la clave inicial) se aplican en las rondas. Las rondas

son un conjunto de operaciones que se aplican para cifrar la información. Con 11 subclaves se generarán 11 rondas.

La primera ronda inicia con clave inicial K_0 y la función *AddRoundKey.* Esta función realiza la suma OR exclusiva (X-OR) entre K_0 y el texto plano.

Las siguientes nueve rondas deben cumplir cuatro funciones cada una y son:

SubBytes: Realiza una sustitución de cada uno de los valores de la matriz de estado utilizando una tabla.

ShiftRows: Realiza una permutación de las filas de la matriz de estado.

MixColumns: Multiplica cada columna de la matriz de estado por un polinomio fijo.

AddRoundKey: Realiza la operación X-OR entre la clave de cada vuelta con la matriz de estado.

En la ronda final se emplea la última subclave K10. Esta ronda consiste de tres funciones: *SubBytes, ShiftRoes* y *MixColumns*. Estas funciones son las mismas que se definieron para las rondas anteriores.

La Figura 1.17 muestra el proceso que sigue cada bloque de texto plano de 128 bits para ser cifrado con una clave de 128 bits en AES.

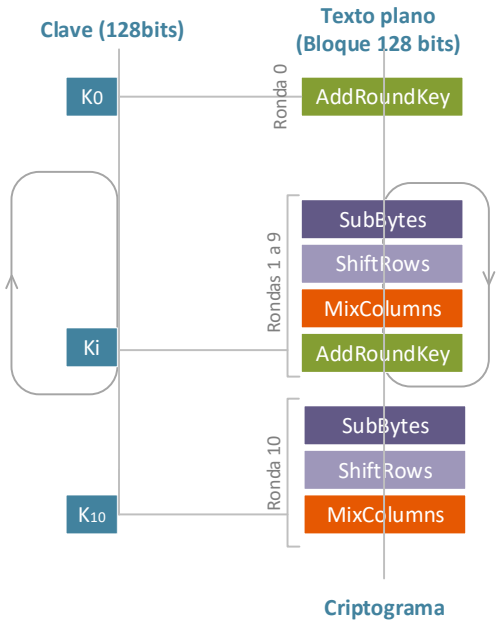


Figura 1.17. Operación de cifrado AES de 128 bits.

El resultado de todo este proceso será un bloque de 16 bytes con el criptograma que resulta de cifrar el bloque de texto plano.

El algoritmo AES tiene distintos modos de operación, pero el más utilizado es CBC (*Cipher Block Chaining*), el mismo que se utilizará en AES para el presente trabajo. Este modo de operación realiza una suma X-OR con el texto cifrado anteriormente, es decir que cada nuevo bloque depende del anterior. Para cifrar el primer bloque de texto plano se requiere de un vector de inicialización IV. La Figura 1.18 muestra el proceso de cifrado y descifrado del modo CBC.

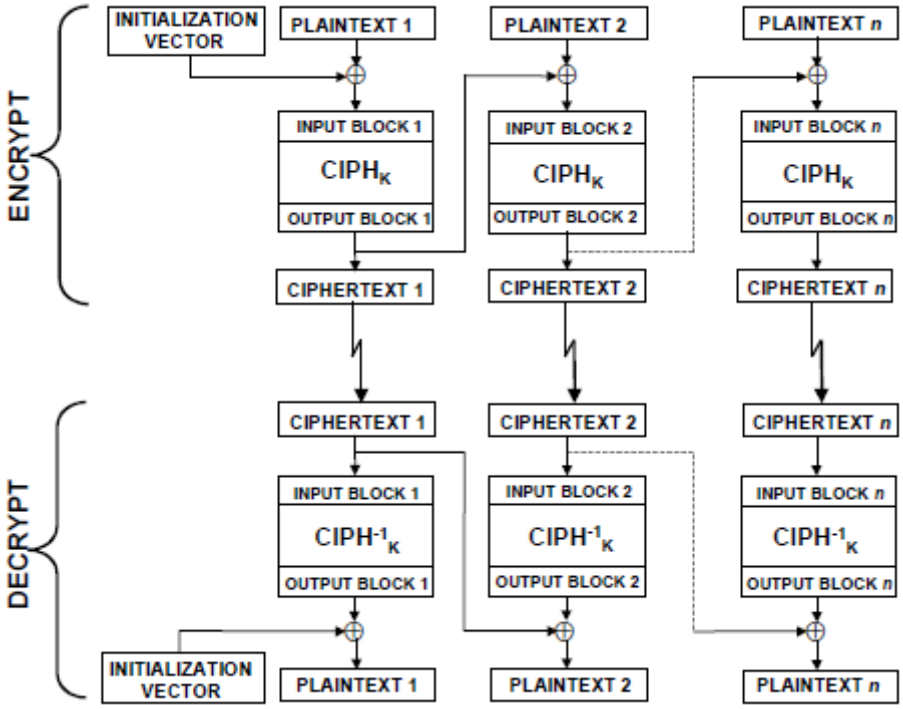


Figura 1.18. Operación del modo CBC [30].

Cuando se han cifrado todos los bloques de información se concatenan los resultados para obtener el resultado final. Para el proceso de descifrado se aplica la función inversa del cifrado como se muestra en la Figura 1.18.

1.3.4.4. Código de Autenticación de Mensajes basado en Cifrado (CMAC)

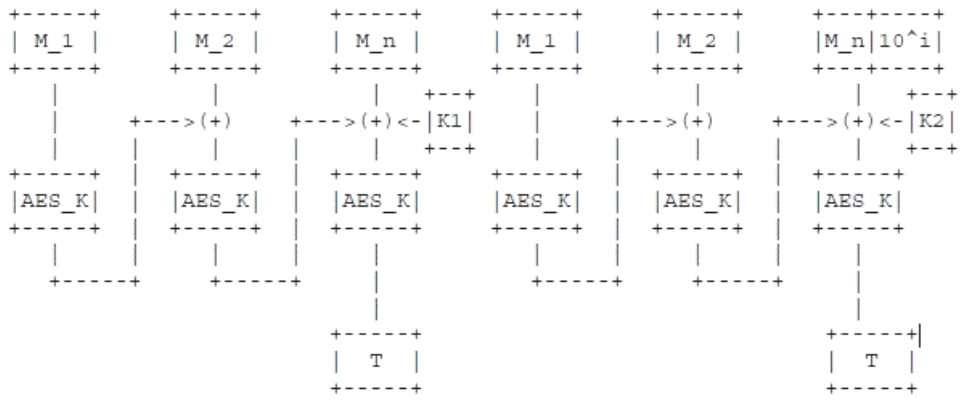
Fue creado por el NIST, el cual especifica un código de autenticación de mensajes que utiliza un cifrado de bloque simétrico como el algoritmo AES. CMAC proporciona servicio de integridad de los datos, ya que es posible detectar las modificaciones intencionales y accidentales de los mismos [28].

Como ya se mencionó, CMAC utiliza un cifrado de bloque simétrico por lo que es posible combinarlo con AES. AES-CMAC está descrito en el RCF-4493 [31].

La salida generada por AES-CMAC es un *string* de 128 bits denominado MAC. Para generar una MAC es necesario tomar una llave secreta, un mensaje de longitud variable y la longitud del mensaje en octetos como elementos de entrada y retornar la MAC [31]. La base de AES-CMAC es el modo CBC-MAC, el cual es un algoritmo MAC basado en el modo CBC que ya se explicó anteriormente.

Si a un mensaje M, se le aplica el modo CBC-MAC, CMAC define dos casos de operación para este modo. El primer caso se aplica para los bloques de mensajes que son múltiplos de 128 bits y el segundo caso cuando el último bloque no es múltiplo de 128 bits. El funcionamiento de los dos casos se muestra en la Figura 1.19.

Ambos casos difieren en su modo de operación en el último bloque, para el caso (a), el último bloque del mensaje realiza una operación X-OR con una subclave K1. Para el caso (b), el último bloque debe ser rellenado con 10^i para realizar una operación X-OR con la subclave K2; donde i es *128-8*r-1* y r es un valor mayor o igual que cero, pero menor que 16.



(a) M múltiplo de 128 bits (b) M no es múltiplo de 128 bits

Figura 1.19. Casos de operación AES-CMAC [31].

Las subclaves K1 y K2 son generadas a partir de la clave principal K del algoritmo AES.

1.3.4.5. Contador con CBC-MAC

CCM está definido para usar cifrado de bloque de 128 bits. Para este trabajo de titulación se utilizará AES como el cifrado de bloque de este algoritmo.

El modo CCM especifica dos parámetros que deben ser escogidos antes de iniciar, los cuales son: el número de octetos en el campo autenticación M y el número de octetos en el campo longitud L.

Posteriormente se definirán los elementos para la autenticación y cifrado del mensaje, estos elementos se mencionan a continuación:

- Una llave K para el cifrado de bloque.
- Un *Nonce* N de 15-L octetos. No se debe duplicar mientras dura el proceso.
- El mensaje m con un total de l(m) octetos, mayor o igual que 0 pero menor que 2^{8L}.
- Datos autenticados adicionales, son opcionales.

CCM define dos tipos de procesos para el mensaje. El primero, la autenticación y segundo el cifrado. En la autenticación se calcula una etiqueta T, haciendo uso del modo CBC-MAC. Para lo cual primero es necesario definir los siguientes bloques:

- B0 está compuesto por los campos: Bandera, *Nonce* y l(m).
- B1…Bn, es el mensaje dividido en bloques de 16 octetos.

Si el último bloque Bn no es múltiplo de 16 Bytes, entonces se lo rellena con ceros. Una vez definidos los bloques, se puede aplicar CBC-MAC como muestra la Figura 1.20.

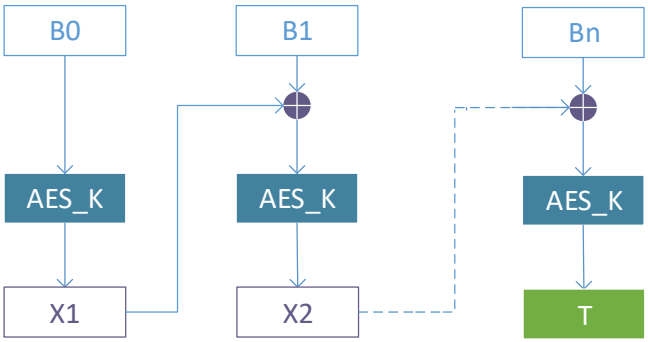


Figura 1.20. Autenticación del modo CCM [2].

La etiqueta T puede ser truncada a un número de bytes determinada si así se requiere.

El cifrado del mensaje se muestra en la Figura 1.21. Este proceso requiere utilizar el modo Contador (CTR) que está formado por los siguientes campos:

- Banderas
- *Nonce* N
- Contador i

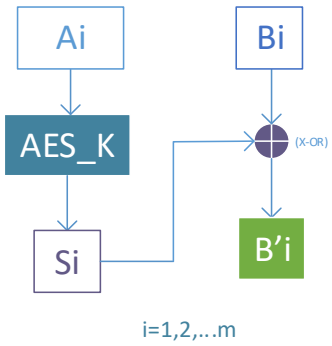


Figura 1.21. Cifrado del modo CCM [2].

Para el cifrado no se requiere que el último bloque tenga relleno en caso de que no sea múltiplo de 16 bytes. Debido a que el ajuste en este caso se da en la longitud del contador, truncándolo a la medida del último octeto. Todos los mensajes B'i obtenidos en cada iteración se han de concatenar para dar como resultado el criptograma. Adicional al cifrado del mensaje, también se cifra la etiqueta T con el valor S0 aplicando la operación X-OR.

1.3.5. FORMATO TRAMAS MAC [2]

Las tramas se encuentran conformadas por campos que se pasan a la capa física a través de los puntos de acceso al servicio, con lo que se permite el intercambio de información entre las entidades a nivel de subcapa MAC. La Figura 1.22 muestra el formato general que tendrán las tramas.

Cabecera MAC	**MAC Frame Body**	**FCS**

Figura 1.22. Formato general trama 802.15.6 [2].

Tanto la cabecera MAC como el MAC *Frame Body* contienen subcampos que se establecerán de acuerdo al tipo de trama que lo requiera. La Figura 1.23 muestra los subcampos contenidos en la cabecera MAC, mientras que la Figura 1.24 indica el formato del MAC *Frame Body*.

Control de Trama	**ID Receptor**	**ID Remitente**	**ID BAN**

Figura 1.23. Formato de la cabecera MAC [2].

Los campos Número de Secuencia de Bajo Orden y *MIC,* solo se encuentran en las tramas: Solicitud de Conexión, Asignación de Conexión, Prioridad de Usuario 0-6 y Emergencia. Estos campos permiten asegurar los mensajes y proporcionar los servicios de seguridad de nivel dos.

Número de Secuencia de Bajo Orden	***Payload***	**MIC**

Figura 1.24 Formato General del MAC *Frame Body* [2].

1.3.5.1. Cabecera MAC

La cabecera MAC cuenta con 7 octetos los cuales se encuentran divididos entre el Control de Trama, ID de Receptor, ID de Remitente y el ID de la BAN (*Body Area Network*). El campo Control de Trama les permite a las entidades involucradas en la comunicación, identificar qué tipo de información se está intercambiando, condiciones de la comunicación, así como aspectos de seguridad, entre otras. El número de bits asignado a cada campo se indica en la Tabla 1.4.

Tabla 1.4 Número de bits de los campos de la cabecera MAC.

Cabecera MAC		**# Bits**
Control de trama	Versión del Protocolo	1
	Política Ack	2
	Nivel de Seguridad	2
	Índice TK	1
	Seguridad BAN/Replay	1
	Sincronización Ack/Primera trama en Tiempo	1
	Subtipo de Trama	4
	Tipo de Trama	2
	Más Datos	1
	Última Trama	1
	Número de Secuencia	8
	Número de Fragmento	3
	Fragmento No-Final	1
	Reservado	4
ID Receptor		8
ID Remitente		8
ID BAN		8

El valor de los campos Subtipo y Tipo de Trama se establece como se muestra en la Tabla 1.5.

Tabla 1.5 Subtipo y Tipo de trama.

SUBTIPO	VALOR SUBTIPO	TIPO	VALOR TIPO
Asociación de Seguridad	0100	Administración	00
PTK	0010	Administración	00
Solicitud de Conexión	0001	Administración	00
Asignación de Conexión	1001	Administración	00
Prioridad de Usuario 0	0000	Datos	01
Prioridad de Usuario 1	1000	Datos	01
Prioridad de Usuario 2	0100	Datos	01
Prioridad de Usuario 3	1100	Datos	01
Prioridad de Usuario 4	0010	Datos	01
Prioridad de Usuario 5	1010	Datos	01
Prioridad de Usuario 6	0110	Datos	01
Emergencia	1110	Datos	01

1.3.5.2. Payload PTK

El *Payload* de la trama PTK se organiza como se muestra en la Figura 1.25. Los valores que corresponden a cada campo se detallan a continuación.

Dirección Receptor	**Dirección Remitente**	***Nonce* Remitente**	**Control PTK**	**PTK_KMAC**

Figura 1.25 *Payload* de la trama PTK [2].

La Tabla 1.6 muestra los campos del *Payload* de la trama PTK con la respectiva cantidad de bits asignada para cada campo.

Tabla 1.6 Número de bits de la trama PTK.

PAYLOAD TRAMA PTK		# Bits
Dirección Receptor		48
Dirección Remitente		48
***Nonce* Remitente**		128
Control PTK	Número de Mensaje	2
	Índice PTK	1
	Reservado	1
	Estado de Creación PTK	4
	Reservado	8
PTK_KMAC		64

1.3.5.3. Payload de las tramas Solicitud de Conexión y Asignación de Conexión

El *Payload* de la trama Solicitud de Conexión y Asignación de Conexión se organizan como se muestra en la Figura 1.26. El número de bits asignado a cada campo se detallan en las Tablas 1.7 y 1.8.

Dirección Receptor	Dirección Remitente	Capacidad MAC	Capacidad PHY	Indicador Cambio de conexión	Tasa de datos Ack solicitado	Periodo de activación solicitado	Fase de activación solicitado

(a)

Dirección Receptor	Dirección Remitente	Modo/Estado	Número de la actual súper-trama	Earliest RAP1 End	EAP2 Start	

	Capacidad MAC	Capacidad PHY	Indicador Cambio de conexión	Tasa de datos Ack asignado	Fase de activación asignado	Periodo de activación asignado

(b)

Figura 1.26. (a) *Payload* de la trama Solicitud de Conexión. (b) *Payload* de la trama Asignación de Conexión [2].

Los campos del *Payload* de las tramas de Solicitud y Asignación contienen subcampos, los cuales se muestran en las tablas que vienen a continuación.

La Tabla 1.7 muestra los campos del *Payload* de la trama Solicitud de Conexión con la respectiva cantidad de bits asignada para cada campo.

La Tabla 1.8 muestra los campos del *Payload* de la trama Asignación de Conexión con la respectiva cantidad de bits asignada para cada campo.

1.3.5.4. Payload de la trama de Datos

La trama de Datos corresponde a la información tomada del cuerpo humano y que es transmitida para su posterior uso. Por lo tanto, el *Payload* de esta trama está conformada únicamente por la información cifrada de acuerdo como lo establece el proceso de Seguridad del Mensaje.

En el Anexo A se presentan las tablas con los valores correspondientes a las tramas PTK, Solicitud de Conexión, Asignación de Conexión y Datos.

Tabla 1.7 Cantidad de bits de la trama Solicitud de Asignación.

PAYLOAD TRAMA SOLICITUD DE CONEXIÓN		# Bits
Dirección Receptor		48
Dirección Remitente		48
Capacidad MAC	CSMA/CA	1
	Acceso Aloha Ranurado	1
	Acceso Poleo Tipo I	1
	Acceso Poleo Tipo II	1
	Acceso Programado	1
	Acceso No Programado	1
	Fragmentación/Reensamble	1
	Tramas de Comando	1
	Nodo Activo Siempre	1
	Aprovisionamiento Tiempo de Guarda	1
	L-Ack/B-Ack	1
	G-Ack	1
	Nodo de Retransmisión	1
	Hub/Nodo de Retransmisión	1
	Beacon Shifting	1
	Salto de Canal	1
	Subtipos de Dato	4
	Reservado	4
Capacidad PHY	Tasa de Datos 0	1
	Tasa de Datos 1	1
	Tasa de Datos 2	1
	Tasa de Datos 3	1
	Tasa de Datos 4	1
	Tasa de Datos 5	1
	Tasa de Datos 6	1
	Tasa de Datos 7	1
Indicador de Cambio de Conexión	Ack Data Rates Change	1
	Cambio Fase de Activación	1
	Cambio Periodo de Activación	1
	Solicitud Uplink	1
	Solicitud Downlink	1
	Solicitud Bilink	1
	Solicitud Bilink No Programado	1
	Cambio de Orden Canal IE	1
Tasa de Datos Ack Solicitado	Control Tasa de Datos del Ack del Nodo	1
	Tasa de Datos del Ack del Nodo	3
	Control de Tasa de Datos del Ack del *Hub*	1
	Tasa de Datos del Ack del *Hub*	3
Periodo de Activación Solicitado		16
Fase de Activación Solicitado		16

Tabla 1.8 Cantidad de bits de la trama Asignación de Conexión.

PAYLOAD TRAMA ASIGNACIÓN DE CONEXIÓN		**#Bits**
Dirección Receptor		48
Dirección Remitente		48
Modo/Estado	Indicador de Acceso	2
	Estado de la Conexión	5
	Reservado	1
Número de la Actual Súper-Trama		8
Earliest RAP1 End		8
EAP2 Start		8
Capacidad MAC	CSMA/CA	1
	Acceso Aloha Ranurado	1
	Acceso Poleo Tipo I	1
	Acceso Poleo Tipo II	1
	Acceso Programado	1
	Acceso No Programado	1
	Fragmentación/Reensamble	1
	Tramas de Comando	1
	Hub Clock PPM	1
	Aprovisionamiento Tiempo de Guarda	1
	L-Ack/B-Ack	1
	G-Ack	1
	Nodo de Retransmisión	1
	Hub/Nodo de Retransmisión	1
	Beacon Shifting	1
	Salto de Canal	1
	Subtipos de dato	4
	Reservado	4
Capacidad PHY	Tasa de Datos 0 a la 7	8
Indicador de Cambio de Conexión	Ack Data Rates Change	1
	Cambio Fase de Activación	1
	Cambio Periodo de Activación	1
	Solicitud Uplink	1
	Solicitud Downlink	1
	Solicitud Bilink	1
	Solicitud Bilink No Programado	1
	Cambio de Orden Canal IE	1
Tasa de Datos Ack Asignado	Control Tasa de Datos del Ack del Nodo	1
	Tasa de Datos del Ack del Nodo	3
	Control de Tasa de Datos del Ack *Hub*	1
	Tasa de Datos del Ack del *Hub*	3
Fase de Activación Asignado		16
Periodo de Activación Asignado		16

2. METODOLOGÍA

En el capítulo uno, se han cubierto aspectos generales de las WBANS y también aspectos específicos del estándar IEEE 802.15.6 que permitan adentrarnos en el objetivo general de este Proyecto de Titulación y cumplir el objetivo general y sus objetivos específicos.

El tema propuesto en este trabajo de titulación requiere que se realice el diseño del software, con la finalidad de permitir su implementación en la siguiente sección. Para llevar a cabo este diseño es necesario recurrir a diagramas UML (*Unified Modeling Language*) para el modelamiento del software.

2.1. FASE DE DISEÑO

2.1.1. ENTORNO OPERATIVO

En el entorno operativo se describe el ambiente sobre el cual el software, que se implementará más adelante, se desplegará para probar su funcionamiento. La Figura 1.1 de la sección anterior, muestra la topología de red que tendrá el entorno de comunicación; para esta topología se debe definir el entorno de operación. Las máquinas que permiten la comunicación tienen los requisitos en software que se muestran en la Tabla 2.1.

Tabla 2.1 Requisitos Software

MÁQUINA	SISTEMA OPERATIVO	ENTORNO DE DESARROLLO	LENGUAJE DE PROGRAMACIÓN
Nodo	Windows 10	Microsoft Visual Studio Community 2019	C#
Hub	Windows 10	Microsoft Visual Studio Community 2019	C#
Equipo	Windows 10	Microsoft Visual Studio Community 2019	C#

Una vez que se ha establecido el software de los componentes de comunicación, en la Tabla 2.2 se especifican las características en hardware que deberán tener las máquinas para el software mencionado en la Tabla 2.1.

Tabla 2.2 Características de Hardware

MÁQUINA	CARACTERÍSTICAS
Nodo *Hub* Equipo	Procesador de 1.8 GHz o superior. RAM de 2 GB mínimo. Recomendado 8 GB. Espacio de almacenamiento mínimo 800 MB.
Conmutador	Conmutador Ethernet capa 2 con mínimo 3 puertos libres.

2.1.2. DIAGRAMAS UML

El Lenguaje Unificado de Modelado UML, es un lenguaje gráfico que permite el diseño y la implementación de sistemas de software. Utiliza un modelado visual para conformar diagramas que permiten visualizar la estructura y comportamiento. UML no es un lenguaje de programación, pero, es útil para generar código en distintos lenguajes. Este lenguaje de modelado tiene especial relación con el análisis y diseño orientado a objetos [32].

Para el modelado del software se utilizarán diferentes diagramas UML que permitan explicar su comportamiento y generar el código para probar el funcionamiento y posterior análisis de seguridad que tiene como objetivo este Trabajo de Titulación.

2.1.2.1. Conexión Cliente-Servidor

El diagrama de secuencia que se muestra en la Figura 2.1, indica cómo se establece la conexión entre las entidades, Nodo y *Hub* con el Servidor, para realizar el intercambio de tramas. Cada entidad se encarga de enviar una solicitud de conexión al Servidor; una vez aceptada, las entidades envían un mensaje para activar un hilo en el Servidor que permita recibir y retransmitir las tramas que son enviadas por el Nodo o *Hub*, posibilitando al Servidor que escuche más peticiones de conexión. A su vez, en las entidades se activa un hilo que permite escuchar los mensajes retransmitidos por el Servidor.

La Figura 2.2 muestra el diagrama de actividades tanto para el cliente como para el Servidor que se ha detallado en el diagrama de secuencia de la Figura 2.1. A la izquierda se encuentran las actividades del Cliente (puede ser Nodo o *Hub*), mientras que a la derecha se visualizan las del Servidor.

2.1.2.2. Proceso de Intercambio de Tramas

En la Figura 2.3 se indica el proceso de intercambio de tramas entre las entidades. El *Hub* envía una trama hacia el Servidor que activará su hilo para el procesamiento de la misma en paralelo a sus funciones; cuando se activa el hilo se ejecuta el método *Escuchar*, para que realice el procesamiento, que en este caso será la retransmisión de la trama a todos los dispositivos conectados.

Cuando el Nodo recibe la trama, éste activa su hilo y ejecuta el método asociado *Escuchar*, que pasa como parámetro la trama recibida al método *Procesar* del Nodo. Cuando la trama se encuentra en este método, se procesa siempre y cuando se cumplan las condiciones de identificación de trama. Una vez aceptada la trama, se genera la siguiente trama que deberá ser enviada al *Hub*.

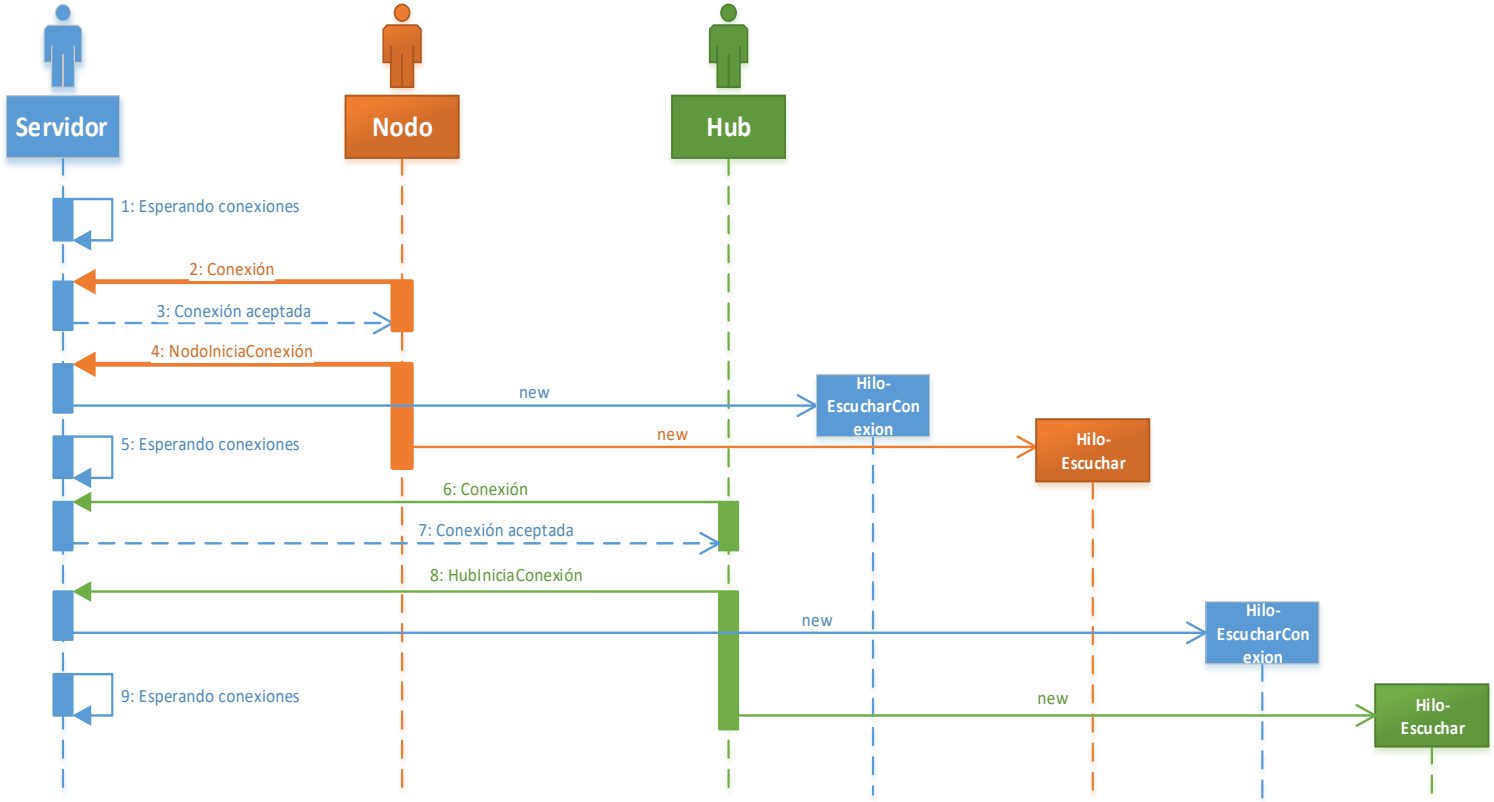


Figura 2.1. Diagrama de secuencia de conexión Cliente-Servidor

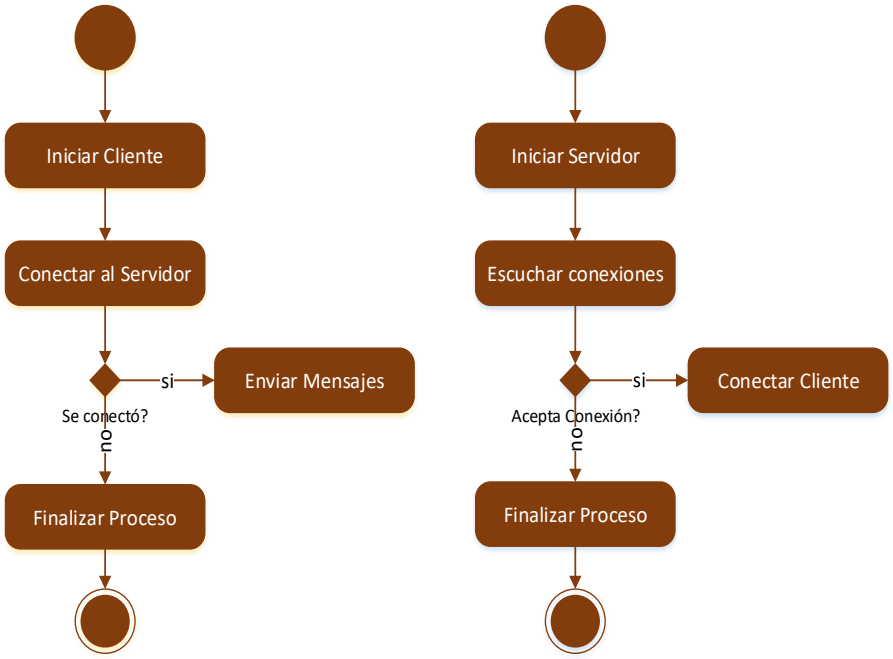


Figura 2.2. Diagrama de actividades de la conexión Cliente y Servidor

La trama que sale del Nodo se envía al Servidor para que se repita el proceso descrito anteriormente, pero ahora la trama la recibirá el *Hub*. El proceso de intercambio de tramas se repetirá hasta que el Servidor o una de las entidades se desconecte o falle.

La Figura 2.4 muestra un diagrama simplificado del proceso de intercambio de mensajes entre los clientes (Nodo y *Hub*) con el Servidor.

2.1.2.3. Intercambio de tramas del paradigma de seguridad

Todas las tramas que se intercambian en los procesos del paradigma de Seguridad, Asociación, Creación PTK y Seguridad del Mensaje, se muestran en la Figura 2.5 de forma simplificada. Además, en esta figura se ha omitido la presencia del Servidor, que es utilizado como elemento de retransmisión de las tramas, debido a que su función dentro del software se explica en la Figura 2.2, y no se hace necesario su presencia en ésta y en ninguna de las siguientes ilustraciones.

Los diagramas de secuencia que reflejan todo el proceso del paradigma de seguridad implementado en el programa, junto con los elementos que se crean para obtener una transmisión con Nivel de Seguridad 2 se adjuntan en el Anexo B.

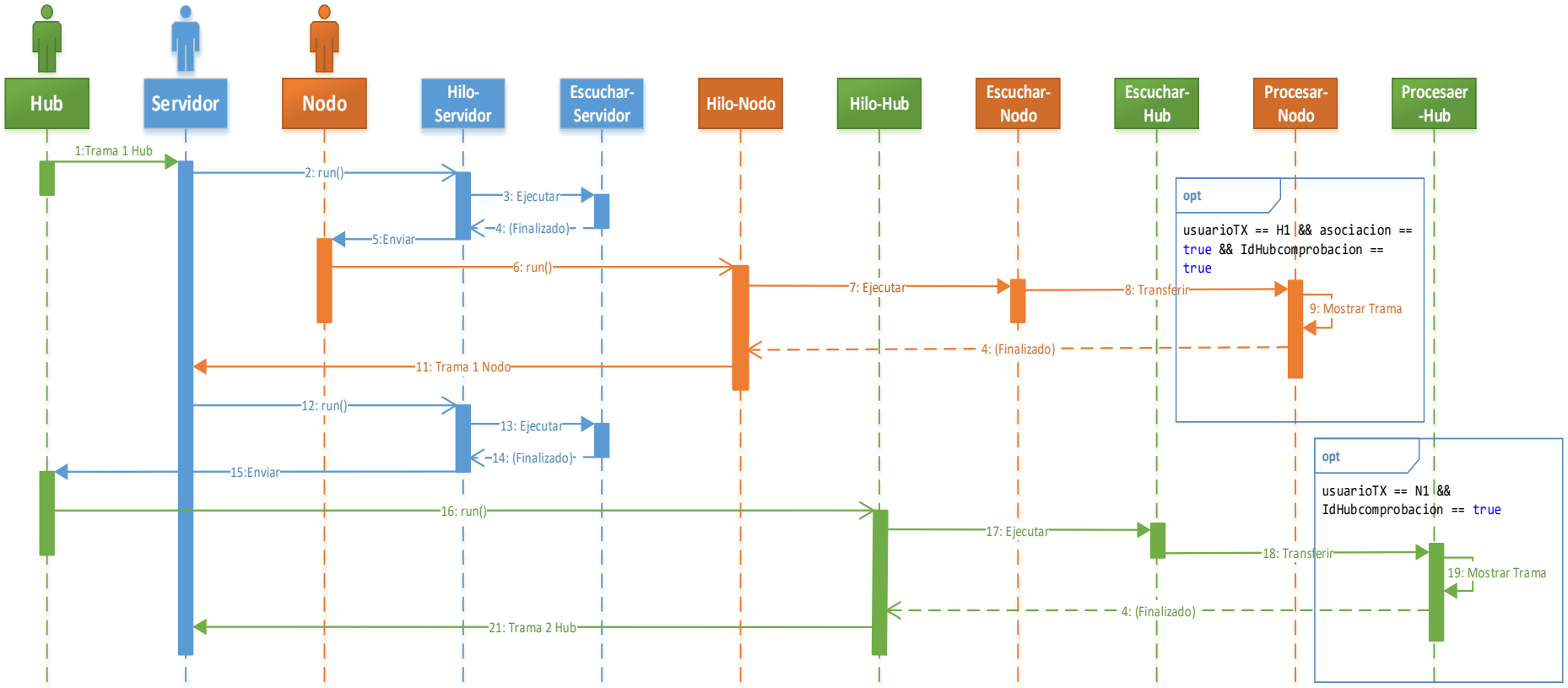


Figura 2.3. Diagrama de secuencia del proceso de intercambio de tramas.

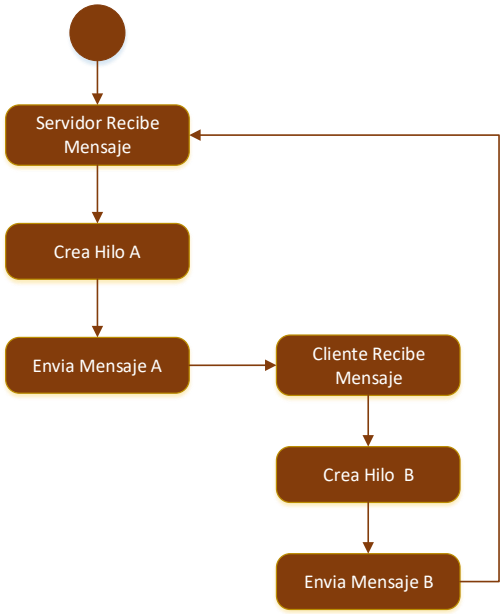


Figura 2.4. Diagrama de actividades proceso de intercambio de tramas.

2.1.2.4. CMAC

En la Figura 2.6 se muestra la secuencia que sigue el software para autenticar las tramas del proceso de Creación PTK mediante la utilización del mecanismo de seguridad CMAC. Para cumplir con el proceso, la entidad manda a llamar a CMAC y éste a su vez solicita la creación de las subclaves K1, K2; utilizando la clave principal MK y una constante definida en [31] es posible la obtención de estos elementos.

Una vez que CMAC ha obtenido las subclaves, la información que debe ser cifrada se divide en bloques de 128 bits. El primer bloque se cifra con AES y la clave principal MK, el resultado es parte de una operación XOR con el segundo bloque. Este segundo resultado se vuelve a cifrar con AES y MK, el proceso se repite hasta antes de llegar al último bloque, donde se realiza una operación XOR con una de las subclaves K1 o K2. La utilización de una u otra clave dependerá si el último bloque lleva relleno o no (K1 se emplea para cuando no existe relleno, mientras que K2 cuando si hay relleno), finalmente este valor se cifra una vez más con AES y la clave MK.

En la Figura 2.7 se encuentran las actividades a seguir para dar cumplimiento al mecanismo de seguridad CMAC. La Figura 2.8 indica cómo se realiza la generación de las subclaves K1, K2 necesarias para CMAC, este proceso se encuentra ya establecido en [31].

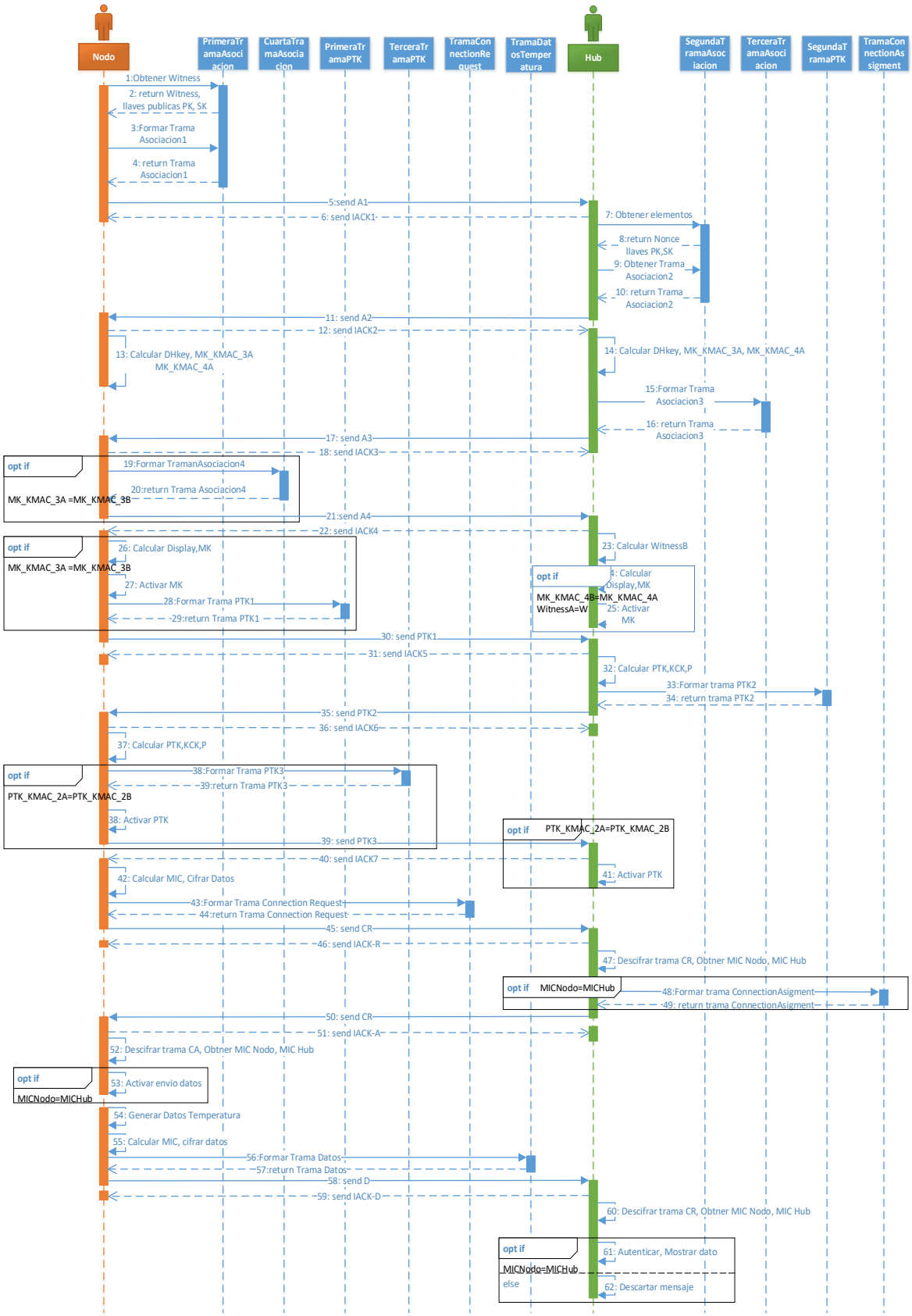


Figura 2.5. Diagrama de secuencia simplificado de intercambio de tramas entre Nodo y *Hub*.

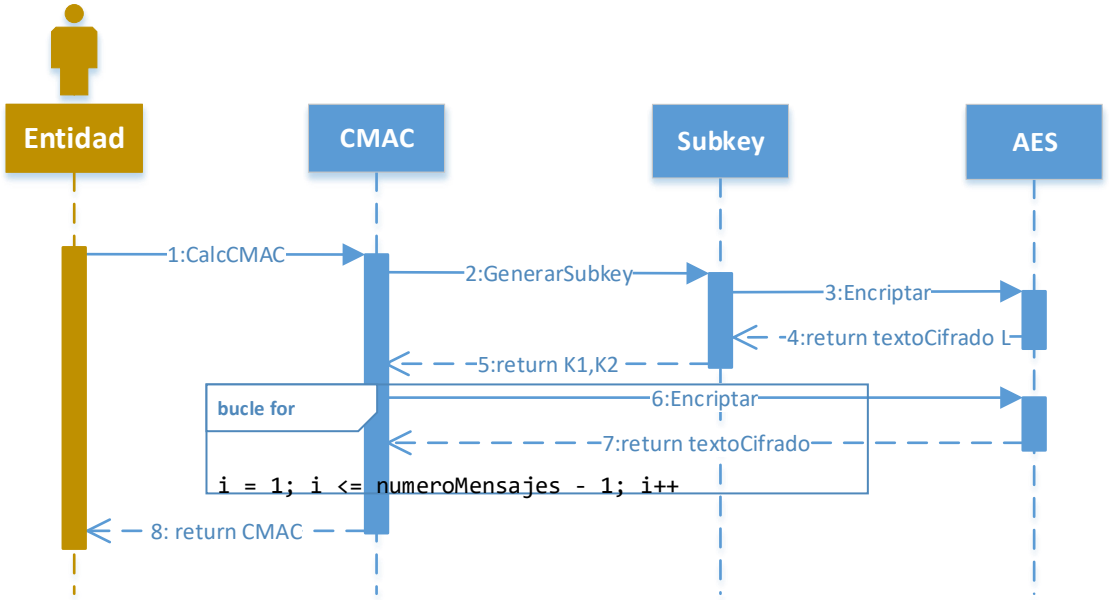


Figura 2.6. Diagrama de secuencia del mecanismo de seguridad CMAC.

2.1.2.5. CCM

Como ya se mencionó, CCM utiliza tanto la autenticación como cifrado para la protección de la información. En la Figura 2.9 se indica la secuencia que debe seguir para cumplir con el algoritmo CCM. En primera instancia, se deberán crear los elementos para la autenticación y cifrado. La Figura 2.10 indica el orden en el que se irán creando los elementos y los datos que se requieren. Las Figuras 2.11 y 2.12 muestran los pasos que ha de seguir el código para crear los elementos de la Figura 2.10.

Cuando se han obtenido los elementos de la Figura 2.10, se puede pasar a obtener la etiqueta de seguridad que permite la autenticación de las tramas. Siguiendo la secuencia de la Figura 2.9, al obtener los elementos se puede llamar a MIC para calcular la etiqueta de seguridad. En este punto se hacen varios llamados a otras instancias que permiten dicho cálculo. Las actividades a desarrollar para obtener la etiqueta se muestran más adelante en las Figura 2.13, y 2.14.

Para finalizar y cumplir con el nivel de seguridad elegido, los datos que se van a transmitir deben ser cifrados; en la secuencia de la Figura 2.9, después de haber obtenido la etiqueta de autenticación se indica cuál es la secuencia que seguirá el proceso de cifrado de los datos. Todas las actividades a desarrollar para dicho proceso se indican en la Figura 2.15.

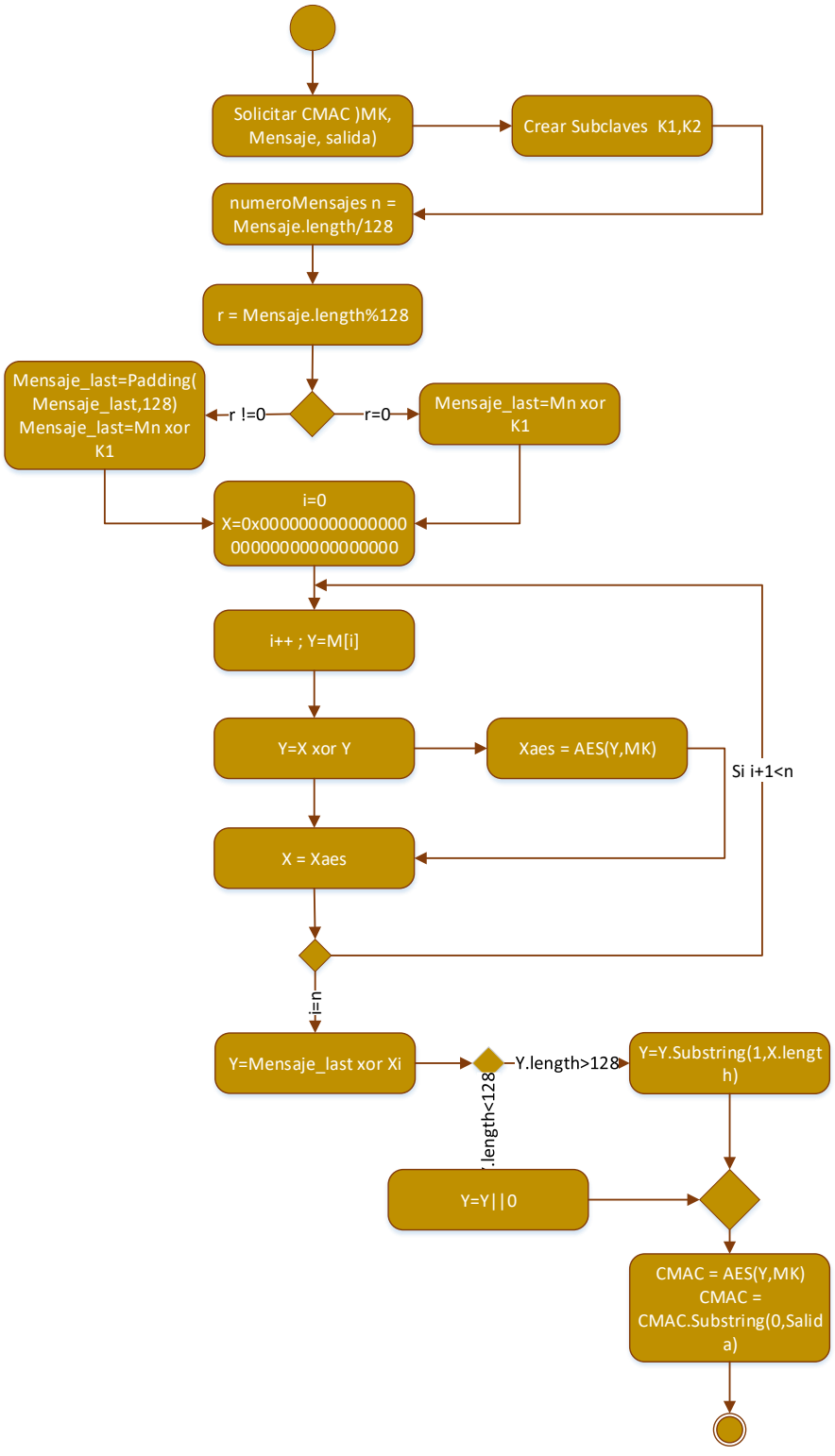


Figura 2.7. Diagrama de actividades para CMAC.

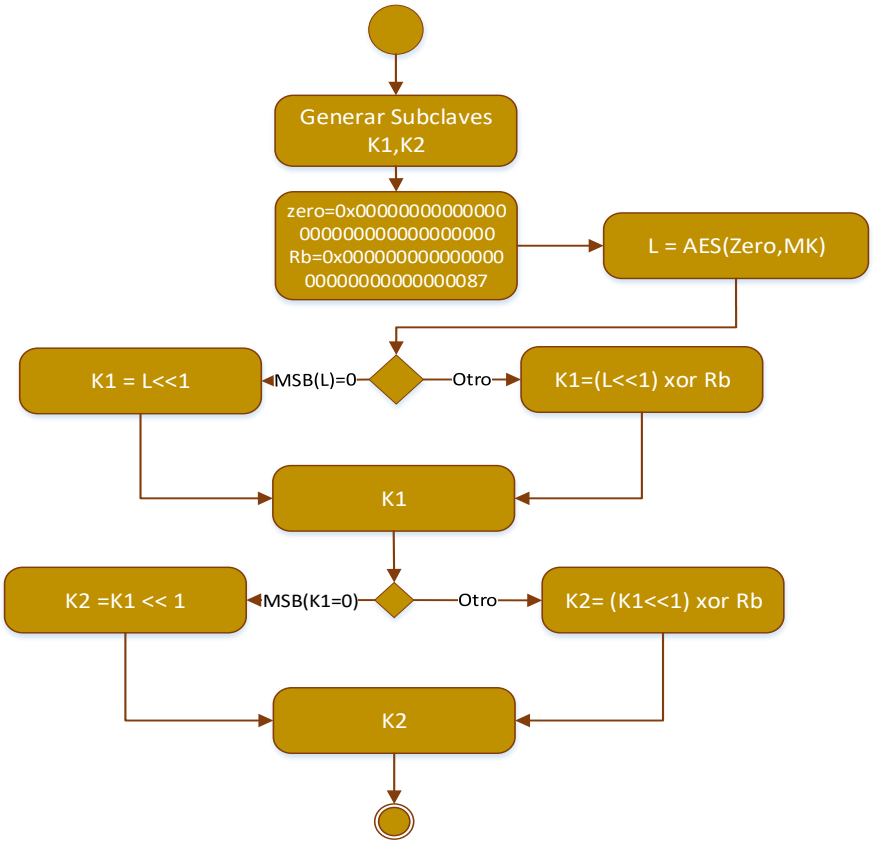


Figura 2.8. Diagrama de actividades para subclaves.

Como se indicó anteriormente, se han de crear algunos elementos y reunir cierta información necesaria para desarrollar el modo de operación CCM como: la Cabecera correspondiente a la trama de Datos, la longitud del dato que está en hexadecimal y el dato, que también se encuentra en hexadecimal. Esta información preliminar permite formar los elementos que vienen a continuación en la Figura 2.10 como son el *Nonce* y los bloques B0 y Ctr0.

El *Nonce* debe ser un elemento único por trama, así lo establece [6], es por ello que utiliza números de secuencia para que no se repita su valor. Utiliza dos números de secuencia, uno de bajo orden con un total de 8 bits, lossn (*Low Order Security Sequence Number*); es decir que puede llegar hasta un valor máximo de 255, y uno de alto orden, hossn (*High Order Security Sequence Number*) con 24 bits a su disposición y por lo tanto un máximo de 2^{24} combinaciones.

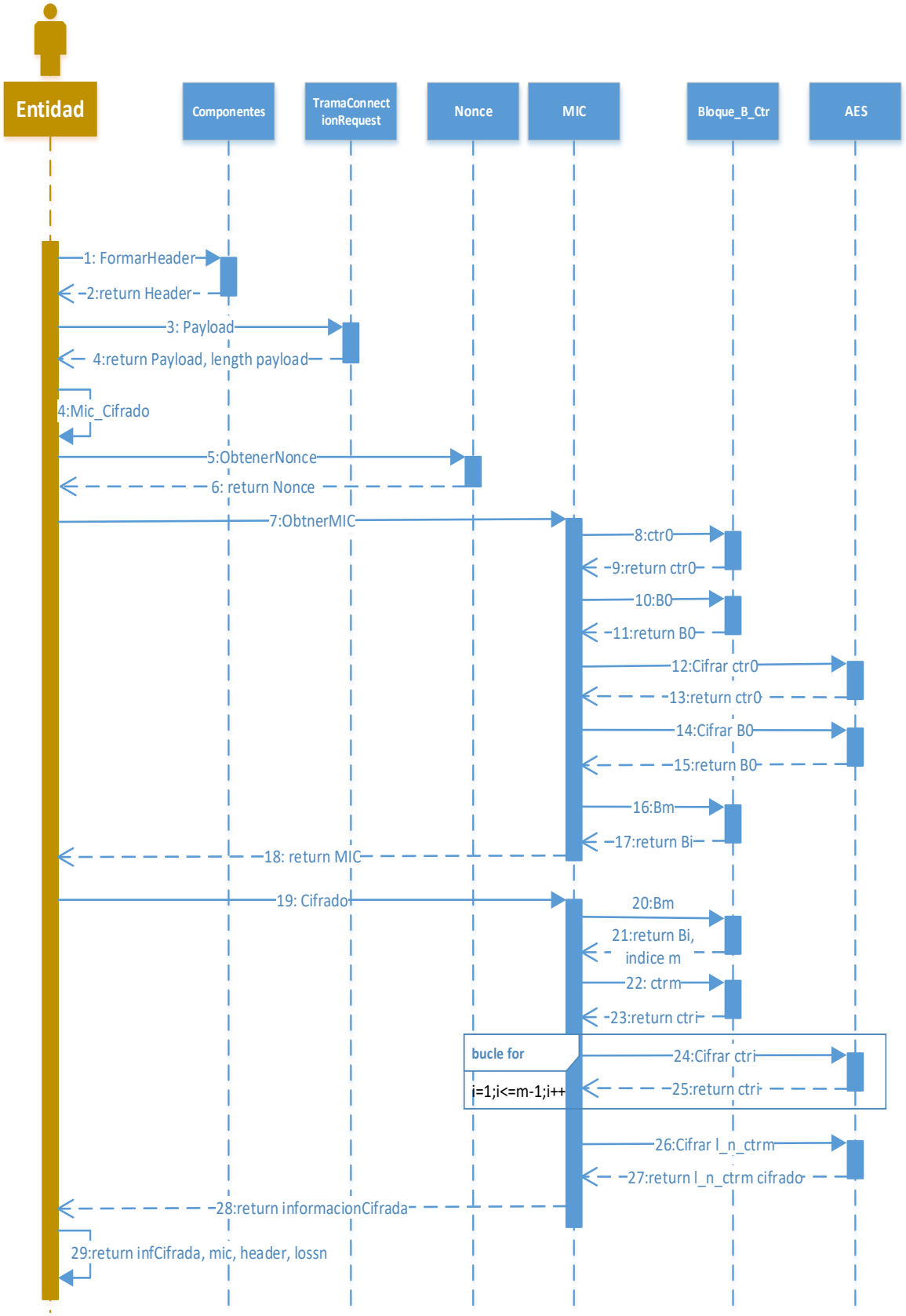


Figura 2.9. Diagrama de secuencia de CCM.

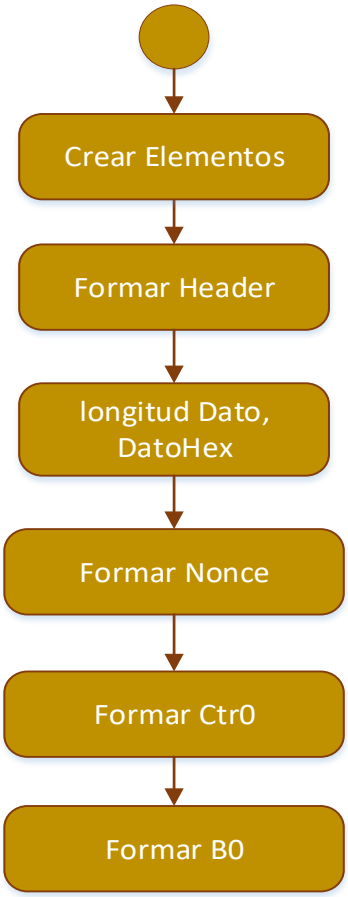


Figura 2.10. Diagrama de actividades para formación de elementos.

La combinación de los números de secuencia de bajo y alto orden permite obtener un total de hasta 2^{32} *Nonces* diferentes. El mecanismo que permite un *Nonce* único utilizando los números de secuencia se muestran en la Figura 2.11. En un principio cada contador de número de secuencia se establece en 0. Como cada trama de información requiere un *Nonce* único, cada vez que se solicite el *Nonce*, ésta irá aumentando su valor de bajo orden (lossn), cuando se llega a su máximo más uno, se establece el contLOSSN en cero y se aumenta en uno el contHOSSN. De esta manera se puede crear un valor único para cada trama de información a ser enviada. Mediante este proceso se evita la duplicación de tramas, ya que este contador se incrementará incluso si la trama se pierde y no llega al *Hub*. Pero como consecuencia después que se ha alcanzado este número de *Nonce*, es necesario volver a calcular una nueva clave para evitar la duplicación del *Nonce* y por ende revelación del mensaje.

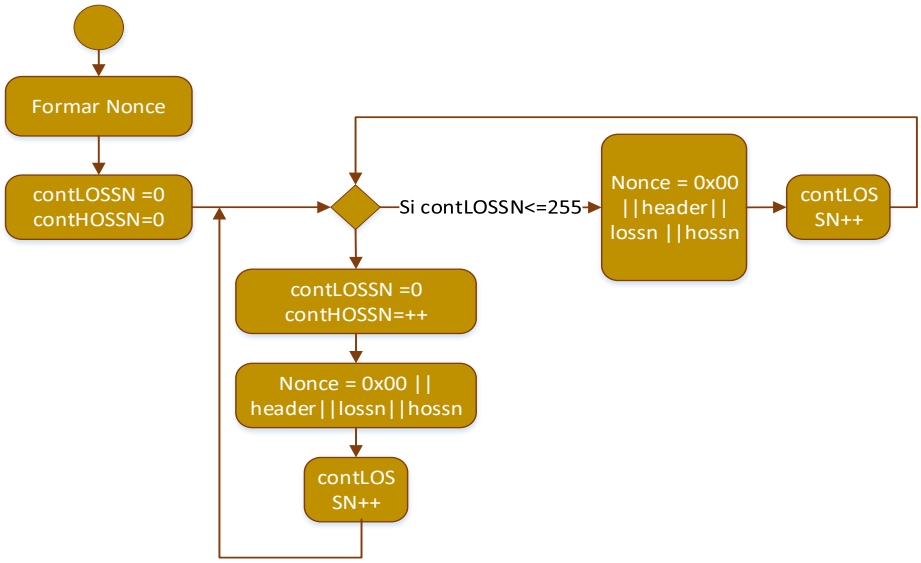


Figura 2.11. Diagrama de actividades para formar el *Nonce*.

Los diagramas de actividades para obtener el Ctr0 y B0 se muestran en la Figura 2.12. Como se puede observar, las formaciones de los dos elementos dependen del *Nonce*. El Ctr0 se forma con el primer valor de un contador, cuyo número depende del número de bloques que se formen a partir del dato, esto se indica más adelante. Mientras tanto, el B0 además del *Nonce*, utiliza la longitud del dato a ser enviado.

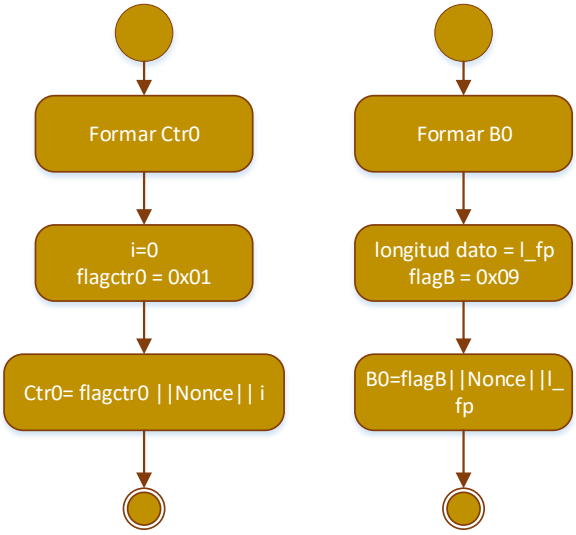


Figura 2.12. Diagrama de actividades para formar el Ctr0 y B0.

Para obtener la etiqueta de seguridad MIC, se debe cifrar con AES los valores Ctr0 y B0 obtenidos y la clave secreta PTK. También es necesario obtener los bloques Bm. Dichos bloques son divisiones del dato en secciones de 16 octetos. Los bloques Bi obtenidos

deberán pasar por una operación XOR con el elemento B0 y con el resultado obtenido en cada iteración, esto sucederá mientras se cumpla la condición. El resultado final de esta iteración junto con el Ctr0 cifrado realizan una operación XOR, y este resultado se trunca al valor de 32 bits u 8 caracteres hexadecimales como lo establece el estándar.

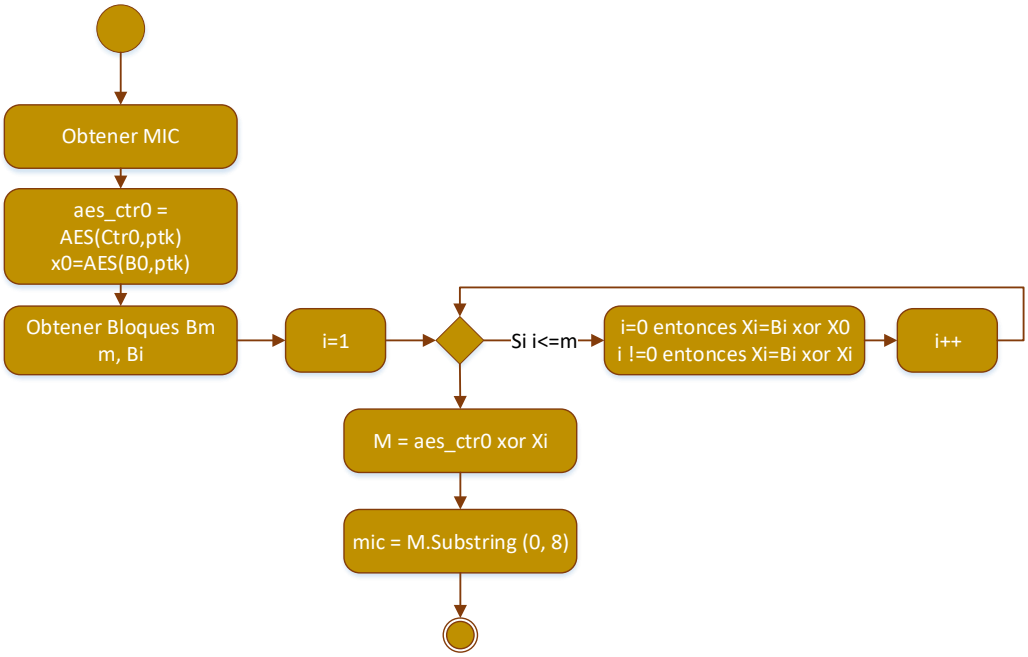


Figura 2.13. Diagrama de actividades para obtención del MIC.

La Figura 2.14 muestra las actividades que deben seguirse para la obtención de los bloques Bm. Los bloques Bm corresponden al dato obtenido, dividido en grupos de 16 octetos, requisito necesario para mandar al cifrador de bloque de 128 bits, AES. Si el dato no es múltiplo de 16 octetos entonces se rellena con ceros al último bloque hasta completarlo, esto para el cálculo de la etiqueta de autenticación. En la figura se utiliza el valor de 32 porque el dato de entrada se trata como una secuencia de hexadecimales. Cada bloque se guarda en una matriz para ser utilizada tanto en el cálculo de la etiqueta MIC como el proceso de encriptación. En el caso de encriptación de los datos, el último bloque no debe ir con relleno, solo con los datos originales.

Las actividades para el cifrado de los datos se muestran en la Figura 2.15. En primer lugar, se requieren de los bloques de datos Bm, obtenidos de acuerdo a lo indicado en la Figura 2.14. Después se procede a calcular los bloques Ctrm. El procedimiento para obtener Ctrm se muestra en la Figura 2.16.

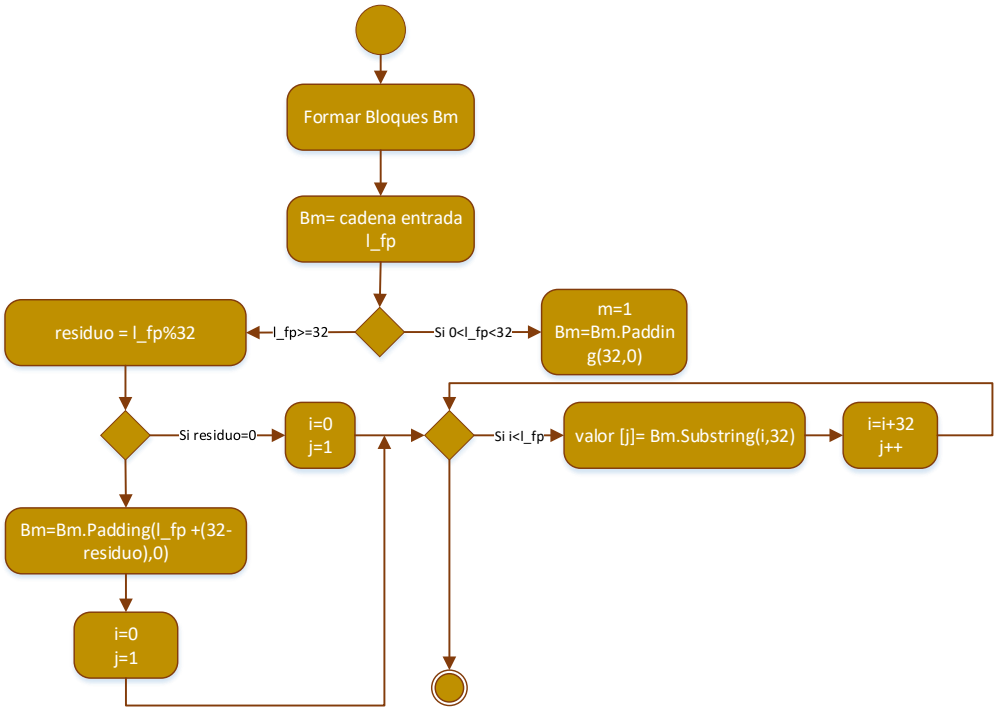


Figura 2.14. Diagrama de actividades para obtener Bloques Bm.

Para poder realizar la encriptación se utiliza un bloque Ctrm por cada bloque bi de datos; la operación que corresponde a realizar entre estos elementos es una operación XOR. Las operaciones de la primera expresión evaluada corresponden a los bloques bi completos. Los valores que se obtienen de cada iteración se guardan en una matriz bi_ que después serán concatenados con el resultado obtenido, al pasar al último bloque por un procedimiento dependiendo de si éste tiene o no relleno de datos.

Una vez que se han encriptado los valores bi_, el siguiente paso es evaluar cuando, el número de bloques m es único y cuando, el último bloque contiene relleno. En el primer caso m=1, se realiza la operación XOR entre el valor bi[m] con su correspondiente bloque ctr[m] cifrado. Es condición que el bloque no lleve relleno si éste no es un múltiplo de 16 octetos.

En el segundo caso, cuando m es mayor que uno, el último octeto puede o no utilizar relleno, por lo que, si lleva relleno se debe quitar y proceder como cuando m=1. El resultado se concatena con los valores obtenidos bi_ calculados y guardados en la matriz. Al final se tiene una secuencia de datos ilegibles que solo pueden ser leídos por la persona que contenga la clave PTK con la que se cifraron los datos.

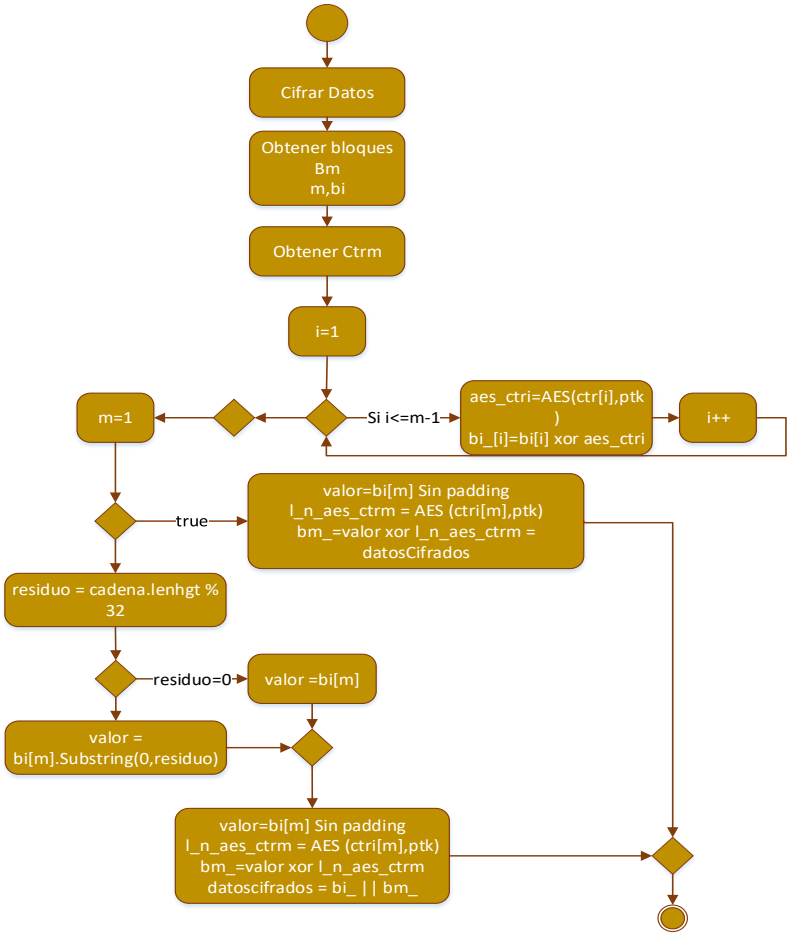


Figura 2.15. Diagrama de actividades para Cifrar Datos.

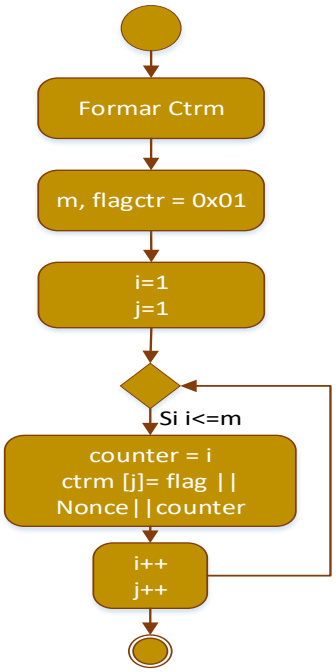


Figura 2.16. Diagrama de actividades para formar Ctrm.

2.1.2.6. Diagrama de clase del Servidor

Para el modelamiento del Servidor, se incluyen las clases Servidor y ConexionCliente. La clase Servidor estará a cargo de escuchar las conexiones que soliciten los clientes y de establecerlas. La clase ConexionCliente se encarga de crear los hilos para los flujos de entrada y salida de los clientes que han establecido conexión con el servidor. La Figura 2.17 muestra las clases necesarias para el funcionamiento del Servidor.

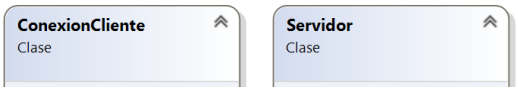


Figura 2.17. Diagrama de clases del servidor.

2.1.2.7. Diagramas de clases

Para el modelamiento del Nodo y el *Hub* se han establecido tres proyectos que permiten dividir a la solución en partes. El primer proyecto Algoritmos, abarca lo que su nombre indica, que son los algoritmos utilizados en el paradigma de seguridad. El segundo proyecto Tramas, contempla la formación de las tramas utilizadas por el Nodo o *Hub*, según corresponda, para ser transmitidas en el paradigma de seguridad. En el proyecto Tramas se incluyen métodos para la formación de la Cabecera de cada trama, obtención de la dirección MAC del Nodo, obtención del CRC, concatenación de los campos para formar las tramas. El tercer proyecto se encuentra la implementación de todas las clases definidas en los proyectos anteriores y la conexión al Servidor.

Algoritmos

Se implementa la clase de cifrado AES presente en todo el proceso del paradigma de seguridad. CMAC para los procesos de Autenticación y Creación del PTK. La clase Subkey es necesaria para la generación de subclaves de K1, K2 utilizadas en CMAC. La clase Curve se utiliza para el proceso de Asociación y permite crear las claves públicas Pk.

Para la autenticación y cifrado de las tramas, se tiene la clase MIC y la clase Bloque_B_Ctr. La primera contiene los métodos que llaman a los elementos que se forman en la segunda clase y que son necesarios para obtener la etiqueta de autenticación y la encriptación de los datos.

Dado que el Nodo recibe una trama asegurada desde el *Hub*, requiere las mismas clases de descifrado que el *Hub* clases las cuales son: Decrypt_Bloque_B_Ctr y DecryptMIC.

Por último, también se define aquí una clase para el cálculo del Código de Redundancia Cíclica (CRC) de las tramas, campo que es parte de todas las tramas que tienen un *Payload* diferente de cero.

La Figura 2.18 muestra el diagrama de clases del proyecto Algoritmos. Este diagrama es común tanto para Nodo como para el *Hub* y todas las clases cumplen las mismas funciones en cada entidad.

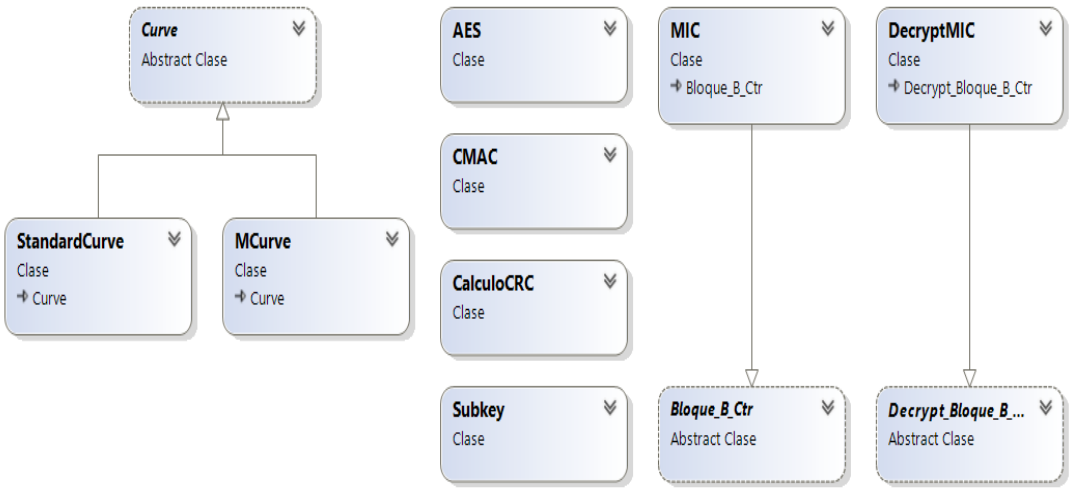


Figura 2.18. Diagrama de clases de Algoritmos.

Tramas

Se implementan clases con las tramas que le corresponden a ser enviadas a cada entidad. Estas clases como se mencionó tienen métodos que permiten la obtención de la dirección MAC, formación del header, llamada al cálculo de CRC y concatenación de todos los elementos para la transmisión de la correspondiente trama. Aquí se incluyen la clase *Nonce*, elemento que pertenece al desarrollo de CCM y la clase Descifrado, que se encarga de obtener la información de la trama entrante, como su *Nonce*, LOSSN, datos cifrados, que serán procesados para poder descifrar y autenticar la trama. Las Figuras 2.19 y 2.20 muestran las clases presentes en las Tramas del Nodo y *Hub* respectivamente.

NodoGUI

En el NodoGUI se encuentra la conexión al Servidor y también los procesos finales para la transferencia de la trama al *Hub*. Además, se establece la salida en el GUI (*Graphical User Interface*) del Nodo, de modo que se observe tanto la llegada de la información como su salida y algunos procesos que se consideran necesarios de ser visualizados.

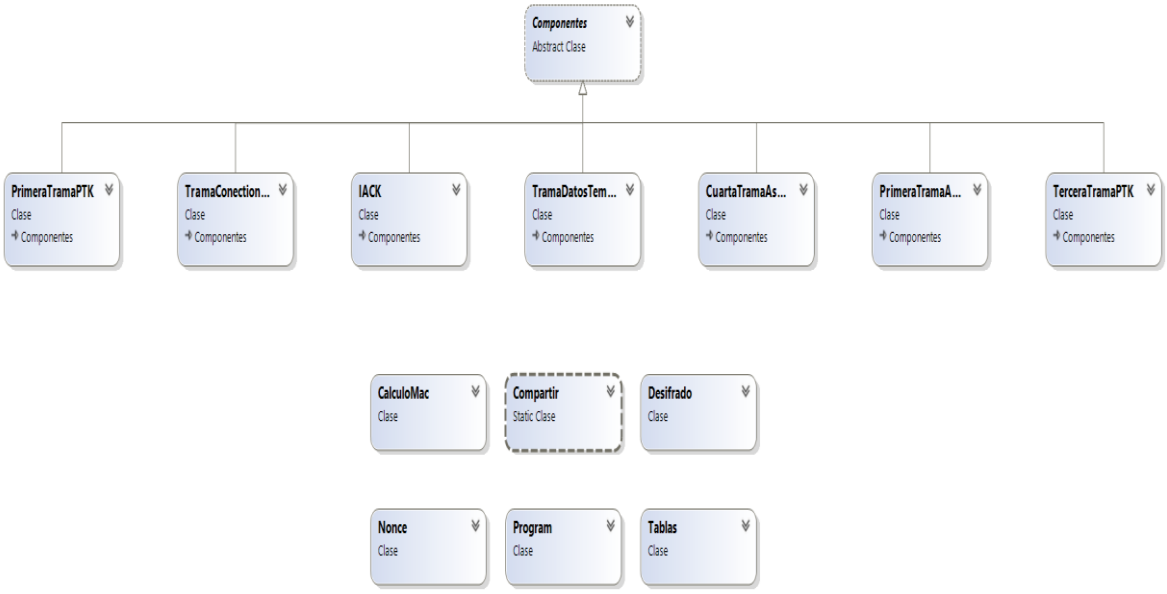


Figura 2.19. Diagrama de clase de Tramas del Nodo.

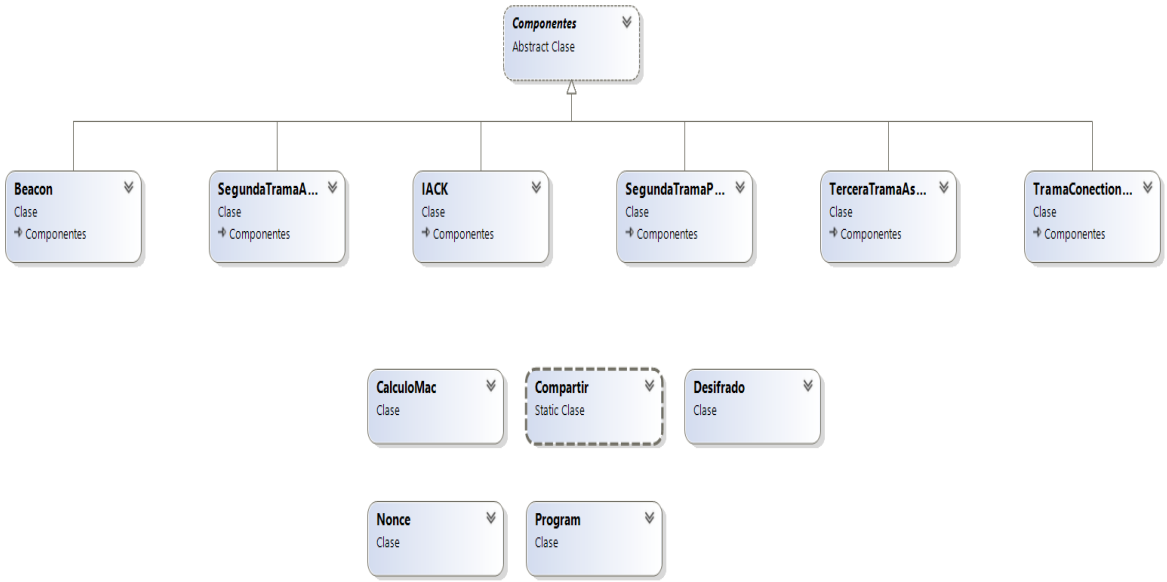


Figura 2.20. Diagrama de clase de Tramas del *Hub*.

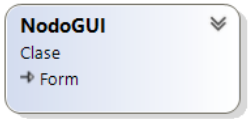


Figura 2.21. Diagrama de clase del NodoGUI.

***Hub*GUI**

Contiene la conexión al Servidor y especifican los procesos finales que se llevarán a cabo para la transmisión de la trama al Nodo; es aquí donde se establece la salida de las tramas y cuál debe ser el proceso que debe activarse al recibir una. Además de encargarse del envío y selección de procesos, se encarga de la visualización de la información de entrada o salida haciéndola visible en la GUI del *Hub*.

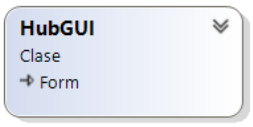


Figura 2.22. Diagrama de clase de *Hub*GUI.

Los diagramas de clases extendidos, con sus atributos y métodos tanto del Nodo como el *Hub* se encuentran en el Anexo C.

2.1.2.8. Interfaz Gráfica de Usuario

La interfaz de usuario tendrá el aspecto mostrado en la Figura 2.23 para el Nodo y la Figura 2.24 para el *Hub*. La sección A de ambas figuras deberá mostrar las tramas recibidas en cada entidad según corresponda. De igual manera, la sección B indicará las tramas que se están enviando desde la correspondiente entidad. La sección C se encargará de mostrar en pantalla algunos elementos que se crean durante cada proceso del paradigma y que son necesarios para indicar sus valores o que se han creado en el proceso. La sección D solo se encuentra en el Nodo, para poder simular la toma de datos de temperatura, activando un botón que permita iniciar la toma de datos y otro que permita detenerlos, con el fin de hacer más manejable y comprensible el software.

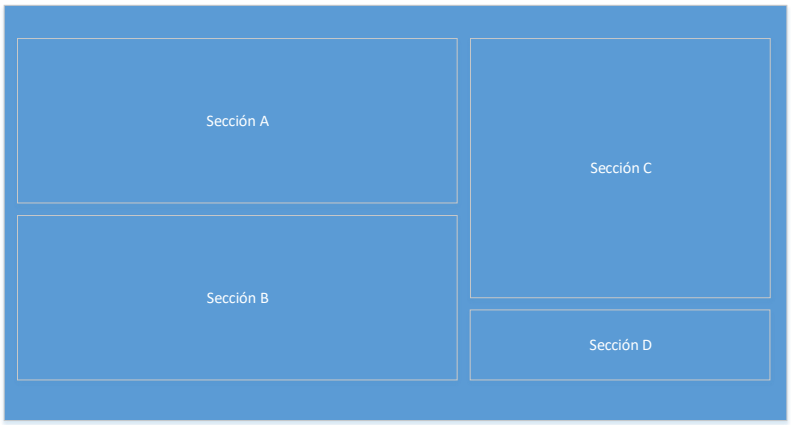


Figura 2.23. Bosquejo GUI del Nodo.

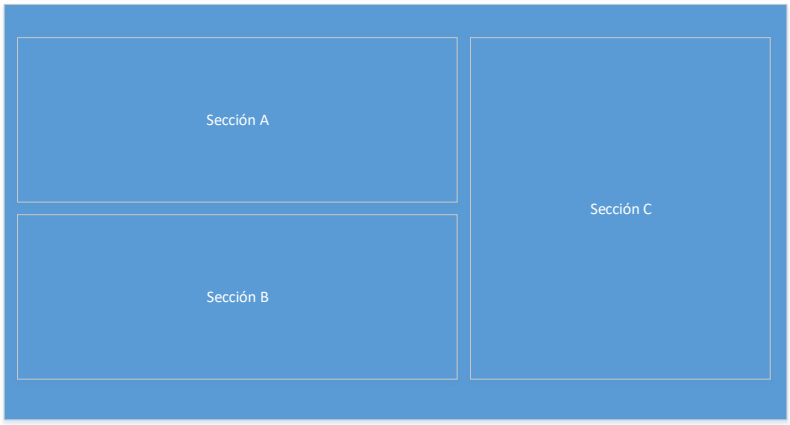


Figura 2.24. Bosquejo GUI del *Hub*.

2.2. FASE DE IMPLEMENTACIÓN

Para la emulación del paradigma de seguridad y su posterior análisis, se convierte el código escrito en Java de [7], al lenguaje de programación propuesto en este proyecto de titulación, C#. Todo el proceso de Asociación desarrollado en [7] se ha tomado como la base para programar este paradigma. Los procesos Creación PTK y Seguridad del Mensaje constituyen la continuación del paradigma y se desarrollan en el presente trabajo de titulación para analizar su seguridad.

En el proceso de Creación PTK, la primera trama transmitida requiere de la obtención de la dirección MAC. Existe una clase que permite obtener dicha dirección y el código implementado se muestra en la Figura 2.25. Busca las interfaces activas, de existir más de una, se selecciona la primera que se liste en el programa y pide obtener la dirección física de dicha interfaz. Una vez que se ha obtenido la dirección MAC, entonces se puede formar la primera trama PTK para ser enviada. La formación de la trama se realiza en la clase PrimerTramaPTK que se encuentra en el Nodo. La clase Mac se encuentra en el Nodo y *Hub* para obtener la dirección física correspondiente y ser utilizada para la formación de las respectivas tramas y en los diferentes cálculos de los procesos.

La Figura 2.26 muestra el código para formar la primera trama PTK, esta clase hereda de la clase Componentes. Se establece un constructor que lleva como atributos los campos de la cabecera MAC de la trama PTK, vistos en el primer capítulo. Cuando se instancie la clase y se pasen valores a estos atributos, se pasarán a la clase Componentes, la cual está encargada de formar la cabecera MAC. El método TramaPTK1 recibe dos parámetros de entrada y solicita la formación de la cabecera de la trama PTK1, la dirección MAC y junto con los otros campos que conforman la trama PTK 1, que se establecen de forma manual,

se concatenan para formar la trama. Por último, se pasa este valor concatenado como atributo al método definido en la clase CalculoCRC, para calcular el CRC de la trama y retornar la trama completa. La formación de las tramas PTK2 y PTK3 se basan en el mismo proceso descrito para la trama PTK1, la información cambiante corresponde a los valores de los campos de la Cabecera y el *Payload*. Los campos de las cabeceras de las tres tramas no cambian, lo que cambia es el valor que contienen.

```
public class CalculoMac
 {
     public String Mac()
     {
         String Mac = "aa";
         try
         {
             Mac = NetworkInterface
            .GetAllNetworkInterfaces()
            .Where(nic => nic.OperationalStatus == OperationalStatus.Up &&
            nic.NetworkInterfaceType != NetworkInterfaceType.Loopback)
            .Select(nic => nic.GetPhysicalAddress().ToString())
            .FirstOrDefault();
         }
         catch (Exception e)
         {
             Console.WriteLine(
                  "\nStackTrace ---\n{0}", e.StackTrace);
         }

         return Mac;
     }
 }
```

Figura 2.25. Código para obtener la MAC.

```
public class PrimeraTramaPTK : Componentes
{
    private string controlPTK;
    private string header;
    private string dirNodo;
    public PrimeraTramaPTK(string pv, string ap, string sl, string ti, string bsr, string af,
                string st, string ty, string md, string lf, string sn, string fn, string nff,
                string rs, string rID, string sID, string bID) : base(pv, ap, sl, ti, bsr, af,
                st, ty, md, lf, sn, fn, nff, rs, rID, sID, bID)
    {        }
    public String TramaPTK1(string dirHub, String nonce_I)
    {
        header = FormarHeader();
        dirNodo = ObtenerDir();
        controlPTK = "1000";
        String ptk_KMAC = "0000000000000000";
        String tramaPTK = String.Concat(header, dirHub, dirNodo, nonce_I, controlPTK, ptk_KMAC);

        tramaPTK = ObtenerCRC(tramaPTK); //envio a que se calcule el crc y se concatene toda la trama
        return tramaPTK;
    }
 }
```

Figura 2.26. Código de la clase PrimeraTramaPTK

La creación del PTK y posterior activación según lo indica el estándar requiere de otros elementos como KCK, P y el PTK_KMAC. La Figura 2.27 muestra la sentencia que permite la obtención de cada uno de estos elementos. En cada uno de ellos se llama a la clase CMAC para que codifique, enviando como parámetros la clave secreta obtenida en el proceso de Asociación MK, el mensaje PTK que corresponde a valores concatenados definidos en el estándar y la longitud que deberá tener el código devuelto.

La creación de elementos se realiza tanto en el lado del Nodo como del *Hub*. Algunos de estos elementos después serán utilizados para compararse y así activar o no la clave PTK.

```
String mensajeKCK = String.Concat(Direccion_Hub, Direccion_nodo, nonce_R, nonceI, ptk_Control);
String mensajeP = String.Concat(Direccion_nodo, Direccion_Hub, nonce_R, nonceI, ptk_Control);

String KCK = CMAC.CalcCMAC(MK_PTK, mensajeKCK, 128);
String P = CMAC.CalcCMAC(KCK, mensajeP, 128);

PTK_KMAC_2A = P.Substring(0, 16);
PTK_KMAC_3A = P.Substring(P.Length - 16, 16);
```

Figura 2.27. Obtención del KCK, P y PTK_KMAC_2.

La Figura 2.28 indica las sentencias empleadas para la creación y activación de la clave PTK en el Nodo, cuando la verificación de los PTK_KMAC se han confirmado. Las mismas sentencias se aplican al Nodo como al *Hub* con los correspondientes valores.

```
// Comprueba SI PTK_KMAC_2A = PTK_KMAC_2B
if (PTK_KMAC_2A.Equals(ptk_kmac_2b) == true)
{

   terceraTramaPTK = new TerceraTramaPTK("0", "10", "00", "0", "0", "0", "0010", "00", "1", "0", "01000000",
   "000",                                    "0", "0000", "01000000", "11000000", "01000000");

   tramaPTK = terceraTramaPTK.TramaPTK3(Direccion_Hub, nonceI, PTK_KMAC_3A);
   tfMensaje = tramaPTK;
   tipoTrama = "P3";
   salidaDatos.WriteLine(usuario + " " + tipoTrama + ": " + tfMensaje);
   salidaDatos.Flush();
   rtbMensajesChat2.AppendText("Trama PTK3: " + tfMensaje + Environment.NewLine + Environment.NewLine);

   String mensajePTK = String.Concat(Direccion_nodo, Direccion_Hub, nonceI, nonce_R, ptk_Control);
   PTK = CMAC.CalcCMAC(MK_PTK, mensajePTK, 128);
   rtbMensajesInf.AppendText("Elemento PTK creado si PTK_KMAC_2A=PTK_KMAC_2B" + Environment.NewLine);
   rtbMensajesInf.AppendText("PTK: " + PTK + Environment.NewLine + Environment.NewLine);

}
else
{
   MessageBox.Show("El proceso de creacion PTK ha fallado - Nodo");
   Application.Exit();
}
```

Figura 2.28 Código para la obtención de la clave PTK.

Uno de los elementos base tanto para el cifrado como la autenticación es el *Nonce*, el cual forma parte del B0 y Ctr. Se define una clase denominada Nonce para la obtención del elemento del mismo nombre y su código se muestra en la Figura 2.29. Se definen dos tipos de contadores que se encuentran en cero para la primera trama de seguridad enviada, y posteriormente se incrementará en uno, cada que se envía una trama. Los números de secuencia son de dos tipos: de bajo orden contLOSSN y de alto orden contHOSSN. El contLOSSN realiza un conteo desde 0 hasta 255, mientras que el contHOSSN tiene un rango más amplio llegando hasta 2^{24}. Estas variables permiten proporcionar un número que no se repita a cada trama enviada.

Los números de secuencia se guardan en una clase estática denominada Compartir, para ser llamados cada que se requiere formar un nuevo *Nonce* obteniendo el último valor asignado y nuevamente guardarlos. El condicional definido ayuda en la incrementación del número de secuencia de alto orden, cuando el número de secuencia de bajo orden llega al límite establecido. Cuando se ha formado el *Nonce* se retorna este valor y se guardan los números de secuencia en la clase Compartir.

```
public String ObtenerNonce(string header)
       {
           contLOSSN = Compartir.ContLOSSN;
           contHOSSN = Compartir.ContHOSSN;

           if (contLOSSN <= 255)
           {
               lossn = Convert.ToString(contLOSSN, 2).PadLeft(8, '0');
               hossn = Convert.ToString(contHOSSN, 2).PadLeft(24, '0');
               lossn1 = new BigInteger(lossn, 2);
               hossn1 = new BigInteger(hossn, 2);

               nonce = String.Concat("0000", header, Compartir.Reverse(lossn1.ToString(16).PadLeft(2, '0')),
                   Compartir.Reverse(hossn1.ToString(16).PadLeft(6, '0'))).ToUpper();

               contLOSSN++;
               Compartir.ContLOSSN = contLOSSN;

               return nonce;
           }
           else
           {
               contLOSSN = 0;
               contHOSSN++;
               lossn = Convert.ToString(contLOSSN, 2).PadLeft(8, '0');
               hossn = Convert.ToString(contHOSSN, 2).PadLeft(24, '0');
               lossn1 = new BigInteger(lossn, 2);
               hossn1 = new BigInteger(hossn, 2);

               nonce = String.Concat("0000", header, Compartir.Reverse(lossn1.ToString(16).PadLeft(2, '0')),
                   Compartir.Reverse(hossn1.ToString(16).PadLeft(6, '0'))).ToUpper();

               contLOSSN++;
               Compartir.ContLOSSN = contLOSSN;
               Compartir.ContHOSSN = contHOSSN;

               return nonce;
           }
       }
```

Figura 2.29. Obtención del *Nonce*.

En la clase Bloque_B_Ctr se encuentran los métodos para obtener los elementos B0, Ctr0, Bm necesarios para el cálculo de la MIC. A continuación, en la Figura 2.30 se indican éstos métodos. B0 y Ctr0 son una concatenación de valores, mientras que Bm requiere de una división del mensaje en bloques de 32, debido a que se está trabajando con hexadecimales. Si la cadena de entrada no es múltiplo de 32, entonces habrá que rellenarla con ceros, ya que es condición necesaria para calcular la MIC.

Los bloques divididos se retornan en una matriz, junto con un índice que indica el número de bloques presentes en la matriz.

```
public Bloque_B_Ctr(int longDato, string nonce, string cadenaHex)
        {
            L_fp = longDato;
            this.Nonce = nonce;
            this.CadenaHex = cadenaHex;
        }
        public String B0()
        {
            l_fp_ = Convert.ToString(L_fp / 2, 16).PadLeft(4, '0');
            b0 = String.Concat(flagB, Nonce, l_fp_).ToUpper();
            return b0;
        }
        public String Ctr0()
        {
            counter = Convert.ToString(i, 16).PadLeft(4, '0');
            ctr0 = String.Concat(flagCtr, Nonce, counter).ToUpper();
            return ctr0;
        }
        public Tuple<string[], int> Bm()
        {
            bm = CadenaHex;
            if (L_fp > 0 && L_fp < 32)
            {
                m = 1;
                bm = bm.PadRight(32, '0');
                valor[m] = bm;
                return Tuple.Create(valor, m);
            }
            else if (L_fp >= 32)
            {
                int residuo = L_fp % 32;
                if (residuo == 0) // es multiplo de 32 no agrego pad
                {
                    for (int i = 0, j = 1; i < L_fp; i = i + 32, j++)
                    {
                        valor[j] = bm.Substring(i, 32);
                        m = j;
                    }
                    return Tuple.Create(valor, m);
                }
                else
                {
                    bm = bm.PadRight(L_fp + (32 - residuo), '0');
                    for (int i = 0, j = 1; i < L_fp; i = i + 32, j++)
                    {
                        valor[j] = bm.Substring(i, 32);
                        m = j;
                    }
                    return Tuple.Create(valor, m);
                }
            }
            else
            {
                Console.WriteLine("No hay mensaje.");
                return null;
            }
```

Figura 2.30. Métodos B0, Ctr0 y Bm.

Las tramas que siguen, una vez terminado el proceso de Creación PTK son aseguradas con cifrado y autenticación. Cuando se instancia la clase MIC se pasan los atributos longitud del dato, *Nonce* y la cadena en hexadecimal para ser establecidos en la clase padre mediante el constructor definido para este propósito, tal como se indica en la Figura 2.31.

```
public MIC(int longDato, string nonce, string cadenaHex) : base(longDato, nonce, cadenaHex)
{ }
```

Figura 2.31. Constructor para acceder a los atributos de la clase base.

La clase MIC permite obtener la etiqueta a través del método mostrado en la Figura 2.32, aquí se llaman a otros métodos que permiten obtener los elementos utilizados para la generación de la etiqueta. Es por ello que la clase MIC hereda de una clase superior nombrada Bloque_B_Ctr, en donde se encuentran los métodos Ctr0, B0, Bm y Ctrm.

```
public string Obtener_MIC(string PTK)
        {
            ctr0 = Ctr0().Remove(0, 2);
            b0 = B0().Remove(0, 2);

            ptk = PTK;
            aes_ctr0 = aes.EncriptAES(ctr0, ptk);
            x0 = aes.EncriptAES(b0, ptk);
            var valoresBm = Bm();
            m = valoresBm.Item2; //es el ultimo indice del array
            bi = valoresBm.Item1;

            Aes_Ctr0 = new BigInteger(aes_ctr0, 16); //cambio a BigInteger
            X0 = new BigInteger(x0, 16);

            for (int i = 1; i <= m; i++)
            {
                Bi = new BigInteger(bi[i], 16);
                if (i == 1)
                {
                    Xi = Bi.Xor(X0);
                }

                else if (i != 1)
                {
                    Xi = Bi.Xor(Xi);
                }
            }
            //AES(Ctr0) x Xm
            M = Aes_Ctr0.Xor(Xi);
            mic = M.ToString(16).Substring(0, 8).ToUpper();   //obtengo los 8 Hex de la izq

            return mic;
        }
```

Figura 2.32. Método para obtención del MIC.

Dentro de la clase Bloque_B_Ctr, se encuentra el método utilizado para la obtención del elemento Ctrm, que junto a los datos Bm, se utilizan para cifrar estos datos. La Figura 2.33 muestra este método.

```
public String[] Ctrm(int m)
        {
            this.m = m;
            for (int i = 1, j = 1; i <= m; i++, j++)
            {
                counter = Convert.ToString(i, 16).PadLeft(4, '0');
                ctrm[j] = String.Concat(flagCtr, Nonce, counter).ToUpper();
            }
            return ctrm;
        }
```

Figura 2.33. Método Ctrm.

El método utilizado para el cifrado de los datos se indica en la Figura 2.34. Cuando se solicita la creación de la etiqueta MIC, se llama a la formación de los bloques Bm, esta variable se guarda y se mantiene para ser utilizada en el cifrado de los datos; es por ello que no se visualiza la llamada al método en esta sección. En el cifrado se utiliza la operación XOR. Esta operación está contenida en la clase *BigInteger*[6] y es necesario que los elementos se encuentren en el mismo formato para realizar esta operación. Es por ello que se puede observar la transformación de *strings* a *BigInteger* para realizar la operación.

El uso de valores *string* y *BigInteger* en el código se debe a que los valores manejados son considerablemente altos y para un manejo más fácil se utiliza estos formatos,

La Figura 2.35 muestra la interfaz gráfica de usuario correspondiente al Nodo, que permite la visualización de las tramas de entrada y salida e información que se va creando como son: claves MK, PTK, etiqueta de autenticación, entre otros elementos. Toda la información adicional se presentará en la sección de Información ubicada a la derecha. Mientras que en la Figura 2.36 se distingue la interfaz del *Hub* con las mismas características de la interfaz del Nodo. Los botones de la GUI del Nodo permiten empezar la transferencia de las tramas o también detener el envío, esto se implementa con la finalidad de dar el control en la transferencia, útil para cuando se realicen las pruebas que se muestran en la sección siguiente.

[6] ***BigInteger:*** es un tipo que puede representar un entero sin signo muy grande.

```
public String Cifrado(string PTK, int longCad)
        {
            ctri = Ctrm(m); //obtengo el array de ctrm desde ctr1
            ptk = PTK; //obtengo el ptk para enviar como Key en AES

            for (int i = 1; i <= m - 1; i++)
            {
                Bi = new BigInteger(bi[i], 16);
                aes_ctri = aes.EncriptAES(ctri[i].Remove(0, 2), ptk);
                AES_Ctri = new BigInteger(aes_ctri, 16);
                bi_[i] = Bi.Xor(AES_Ctri).ToString(16).ToUpper();
            }

            if (m == 1)
            {
                valor = bi[m].Substring(0, longCad); //obteng el ultimo bloque Bm sin el padding, si hay
                Bm_ = new BigInteger(valor, 16);
                l_n_aes_ctrm = new BigInteger(aes.EncriptAES(ctri[m].Remove(0, 2), ptk).Substring(0,
                valor.Length), 16);
                bm_ = Bm_.Xor(l_n_aes_ctrm).ToString(16).ToUpper();
                datosCifrados = bm_;
            }
            else
            {
                int residuo = longCad % 32;

                if (residuo == 0)
                {
                    valor = bi[m];
                }
                else
                {
                    valor = bi[m].Substring(0, longCad % 32);
                }

                Bm_ = new BigInteger(valor, 16);
                l_n_aes_ctrm = new BigInteger(aes.EncriptAES(ctri[m].Remove(0, 2), ptk).Substring(0,
                valor.Length), 16);
                bm_ = Bm_.Xor(l_n_aes_ctrm).ToString(16).ToUpper();
                datosCifrados = String.Join("", bi_);
                datosCifrados = String.Concat(datosCifrados, bm_);
            }

            return datosCifrados;
        }
```

Figura 2.34. Método para el cifrado de datos.

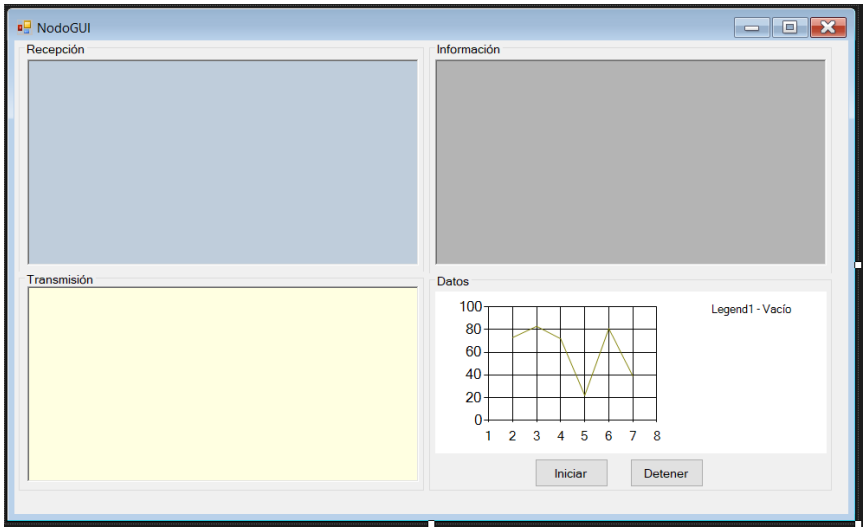


Figura 2.35. GUI Nodo

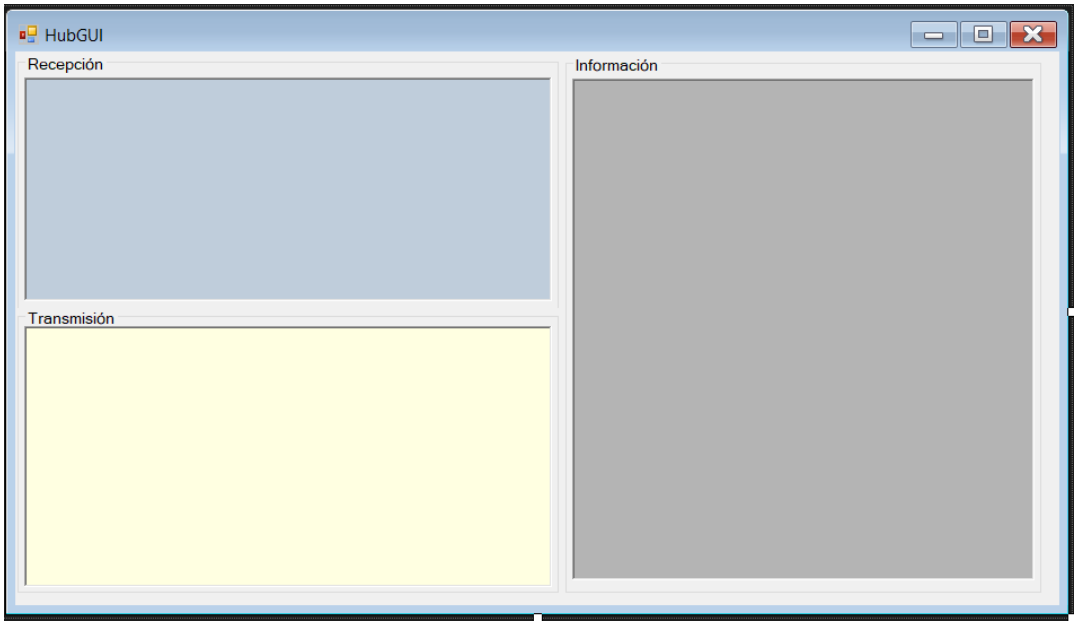


Figura 2.36. GUI *Hub*.

3. RESULTADOS Y DISCUSIÓN

En esta sección se presentarán las pruebas realizadas y sus resultados, lo cual involucra las siguientes fases:

- Pruebas de funcionamiento del Paradigma de Seguridad.
- Prueba de concepto aplicada al mecanismo de seguridad CMAC utilizado en la fase 2, correspondiente al Proceso de Creación PTK.
- Prueba de concepto aplicada al mecanismo de seguridad CCM utilizado en la fase 3, correspondiente a la Seguridad del Mensaje.
- Análisis del nivel de seguridad que existe en cada una de las fases analizadas.

El entorno de emulación de los procesos del Paradigma de Seguridad se presenta en la Figura 3.1 e incluye los siguientes elementos de hardware:

- Dos computadores de escritorio.
- Un conmutador de capa 2 para establecer la comunicación.

Mientras que los elementos de software necesarios son los siguientes:

- Entorno de desarrollo Microsoft Visual Studio para Windows.
- Direccionamiento IP, 192.168.0.1 para el Nodo y 192.168.0.2 para el *Hub*-Servidor.
- El código que contiene el desarrollado Paradigma de Seguridad, así como aquel que contiene al Servidor para la comunicación entre las entidades.

El escenario de la Figura 3.1 permitirá probar el correcto funcionamiento de los procesos y elementos que conforman el Paradigma de Seguridad como lo define el estándar, considerando los límites establecidos en cuanto a funcionamiento de las capas PHY y MAC. Para una exitosa emulación, se deben tener en cuenta las siguientes consideraciones:

- La solución que contiene al Servidor deberá ser implementada en primera instancia para que pueda recibir peticiones de conexión de las entidades y estar en la capacidad de transferir las tramas que se intercambien.
- Las soluciones de las entidades se ejecutan posteriormente en cualquier orden. Se deberán conectar a través de los *sockets* creados mediante TCP.

- Una vez establecida una comunicación exitosa entre el servidor y las entidades correspondientes se empezará con el intercambio de tramas, que serán visibles a través de las GUI en cada entidad, hasta completar las tres etapas del Paradigma de Seguridad.

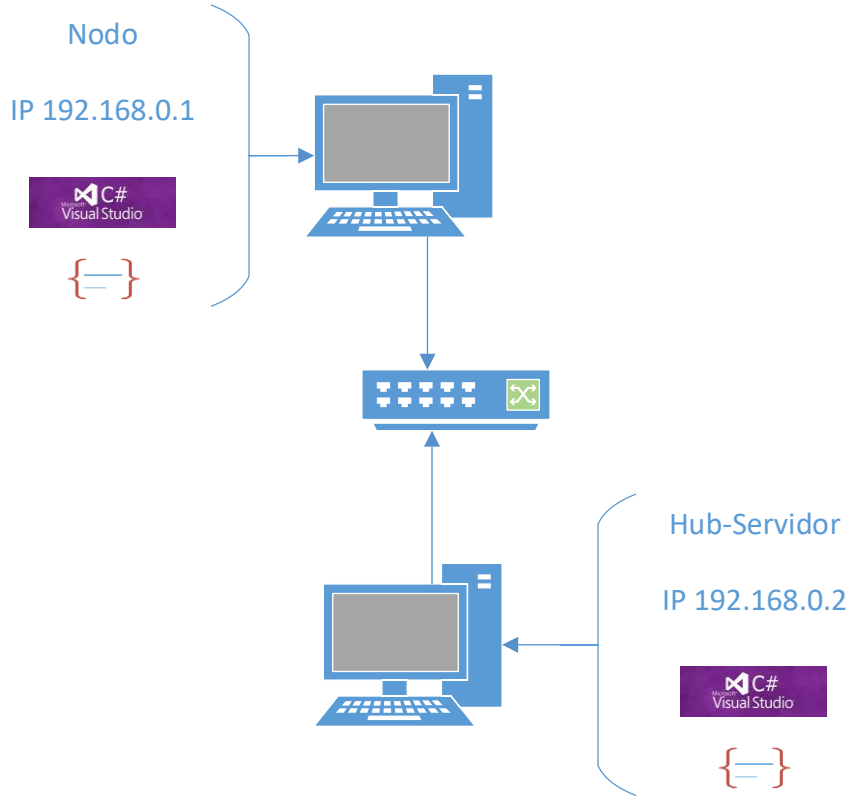


Figura 3.1. Escenario de emulación del Paradigma de Seguridad.

Se mantienen identificadores de entidades y de tipos de trama que permiten el procesamiento, tanto en las entidades como en el Servidor, las cuales se definen a continuación:

- N1: Nodo
- H1: *Hub*.
- IC: Inicio de Conexión.
- BB: *Broadcast Beacon*.
- A1: Primera Trama de Asociación.
- A2: Segunda Trama de Asociación.
- A3: Tercera Trama de Asociación.
- A4: Cuarta Trama de Asociación.
- P1: Primera Trama PTK.

- P2: Segunda Trama PTK.
- P3: Tercera Trama PTK.
- CR: Trama Solicitud de Conexión.
- CA: Tramas Asignación de Conexión.
- DA: Trama de Datos.
- I1: Primera Trama I-Ack.
- I2: Segunda Trama I-Ack.
- I3: Tercera Trama I-Ack.
- I4: Cuarta Trama I-Ack.
- I5: Quinta Trama I-Ack.
- I6: Sexta Trama I-Ack.
- I7: Séptima Trama I-Ack.
- IR: Trama I-Ack en respuesta a la trama CR.
- IA: Trama I-Ack en respuesta a la trama CA.
- ID: Trama I-Ack en respuesta a la trama DA.

Para las pruebas de concepto será necesario la inserción de una tercera entidad no autorizada a la cual se la llamará elemento Equipo A, la cual, al igual que en una comunicación entre entidades autorizadas, deberá cumplir las siguientes condiciones:

- Deberá establecer una conexión con el Servidor, verificado los mensajes en el Servidor.
- Al igual que con las entidades autorizadas, se tiene un identificador para el elemento de Equipo A, que se define a continuación:
  - HA: Elemento de prueba que ejecuta un proceso como *Hub*.
  - NA: Elemento de prueba que ejecuta un proceso como Nodo.

Los identificadores de trama para el elemento de Equipo A se mantienen como los mencionados anteriormente.

La Figura 3.2 muestra cómo estará conformado el escenario para la realización de pruebas de concepto y consecuentemente el análisis para determinar la seguridad de las fases 2 y 3 del Paradigma de Seguridad. Las características en cuanto a software son las mismas, exceptuando la dirección IP del elemento de Equipo A, que deberá ser diferente a las utilizadas en el Nodo y en el *Hub*.

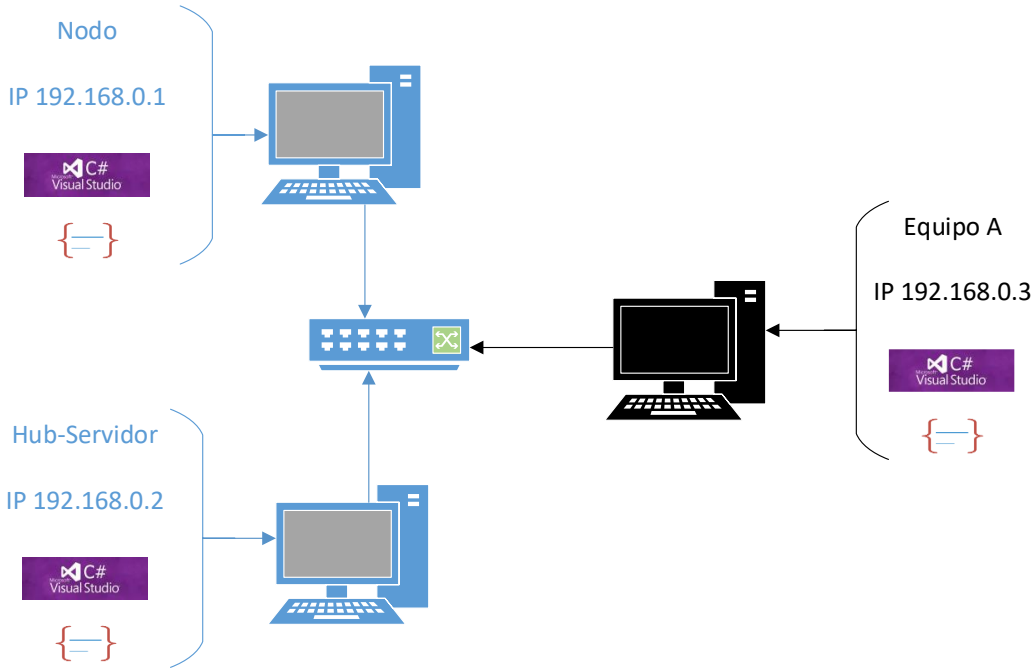


Figura 3.2. Escenario de emulación para la realización de pruebas de concepto, haciendo uso de un Equipo A.

3.1. PRUEBA DE FUNCIONAMIENTO DEL PARADIGMA DE SEGURIDAD

Para llevar a cabo la prueba de funcionamiento se deberá tomar en consideración lo mencionado en cuanto a software, hardware y al escenario de comunicación.

En primera instancia se activará el Servidor, cuya ejecución exitosa se podrá evidenciar mediante consola a través de un mensaje que confirme que el Servidor está activo y listo para recibir conexiones, como se muestra en la Figura 3.3.

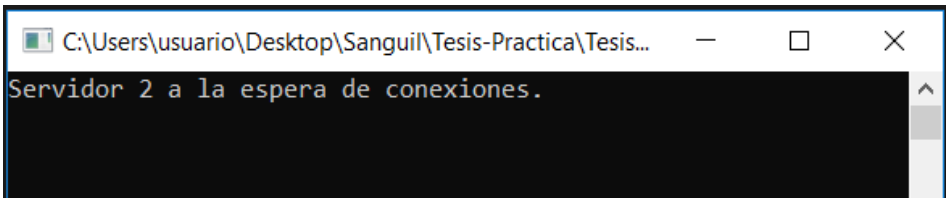


Figura 3.3. Mensaje de consola del Servidor.

Una vez realizado este paso con éxito, se pueden ejecutar a las entidades autorizadas como son el Nodo y *Hub*. Como se ya se mencionó anteriormente en este mismo capítulo,

el orden de la ejecución no es relevante, pero para fines prácticos se iniciará primero el *Hub* y después el Nodo. Como se muestra en la Figura 3.4, la consola del Servidor deberá indicar las conexiones que está recibiendo, mostrando la dirección IP y el puerto por donde se está conectando. El servidor siempre estará disponible para recibir peticiones de conexión incluso si todos los procesos han finalizado. Otra característica que resulta necesario mencionar del Servidor es la retransmisión de mensajes a todos los elementos conectados, incluso aquellos por donde salió el mensaje. Esto no resulta en un problema, ya que gracias a los identificadores de entidad cada una toma el mensaje que le corresponde y el resto lo desecha.

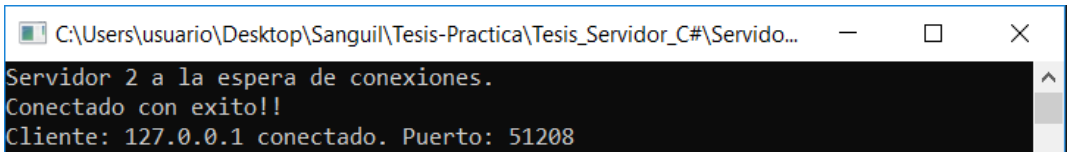


Figura 3.4. Conexión de una entidad al Servidor.

Una vez establecida la conexión *Hub*-Servidor, el *Hub* automáticamente empezará a enviar tramas *Beacon*, como se puede observar en la Figura 3.5, donde se indica la llegada de las tramas *Beacon* al Servidor, para que posteriormente cuando el Nodo se conecte reciba dichas tramas de administración y pueda empezar la primera fase del Paradigma de Seguridad definido como Asociación. El código en el Nodo está desarrollado para que a la segunda trama *Beacon* detectada inicie con el proceso de Asociación. La Figura 3.6 muestra la GUI del Nodo donde se evidencia la llegada de tramas *Beacon* que darán paso a la primera fase del Paradigma de Seguridad.

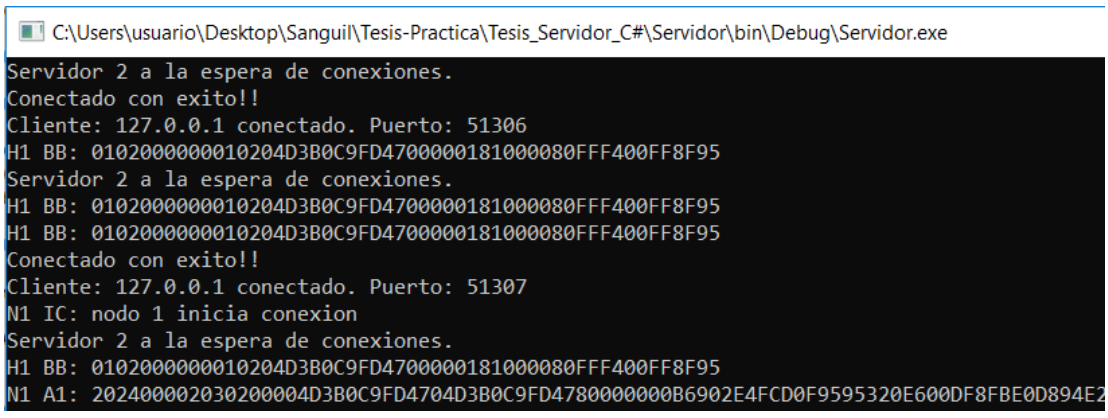


Figura 3.5. Llegada de las tramas *Beacon* al Servidor y conexión de las entidades.

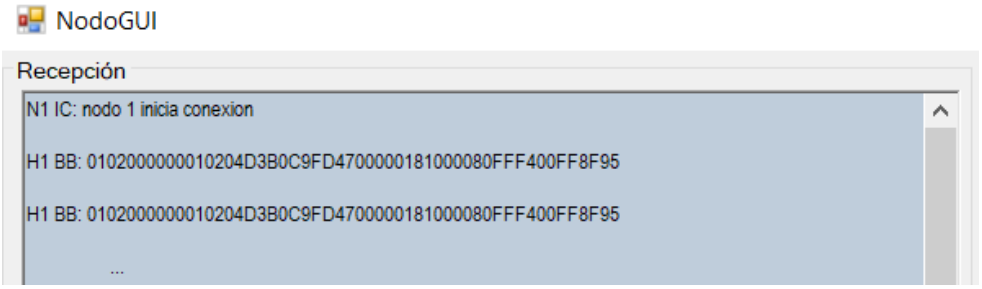


Figura 3.6. Llegada de la trama *Beacon* al Nodo.

3.1.1. PROCESO DE ASOCIACIÓN

Una vez que se ha establecido una conexión exitosa entre las entidades y el Servidor, se inicia el proceso de Asociación. El *Hub* envía las tramas Beacon, y cuando el Nodo las recibe, se activa el proceso de Asociación y calcula los primeros elementos de este proceso como son: llaves públicas a partir de la privada, *Nonce*, el testigo Witness, necesarios para armar la primera trama. En la Figura 3.7 se muestra la creación de estos elementos mencionados. La Figura 3.8 muestra la trama de Asociación *A1* que se transmite desde el Nodo, una vez que se han calculado los elementos.

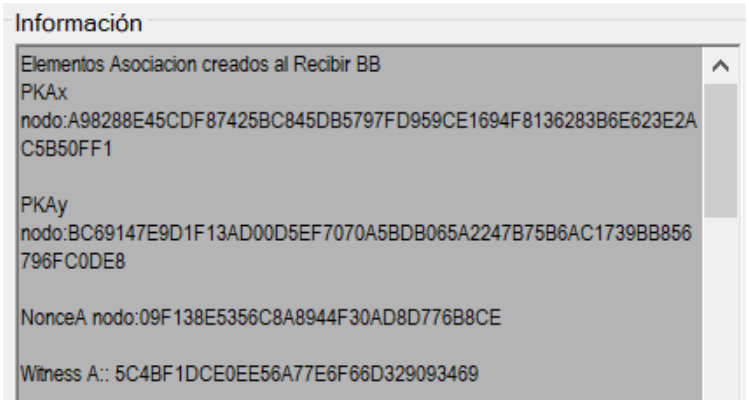


Figura 3.7. Creación de elementos en el Nodo.

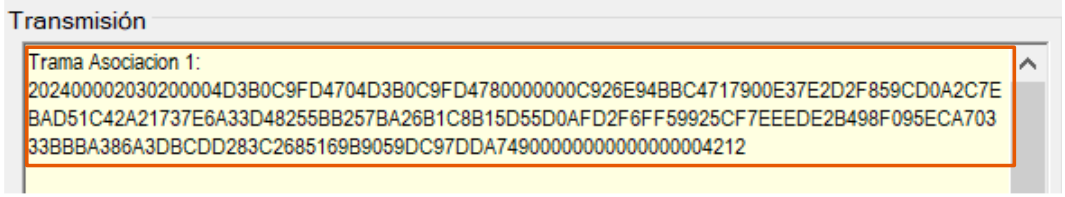


Figura 3.8. Transmisión de la trama A1 desde el Nodo.

El *Hub* recibe la trama A1 enviada por parte del Nodo, la cual se muestra en la sección de recepción de la GUI del *Hub*; como respuesta a la A1 se envía una trama IACK-1 y se calculan el par de llaves pública y privada y el *Nonce* que forman parte de la segunda trama a ser transmitida.

Cuando se transmite la A2, el Nodo envía un IACK-2, y al recibirla el *Hub*, procede elementos como Temp1 y los MK_KMAC. Después forma la trama A3 y la envía al Nodo; en respuesta el Nodo transmite una trama IACK-3. Las Figuras 3.9, 3.10 y 3.11 muestran el proceso descrito anteriormente, el cual se desarrolla solo en el *Hub*.

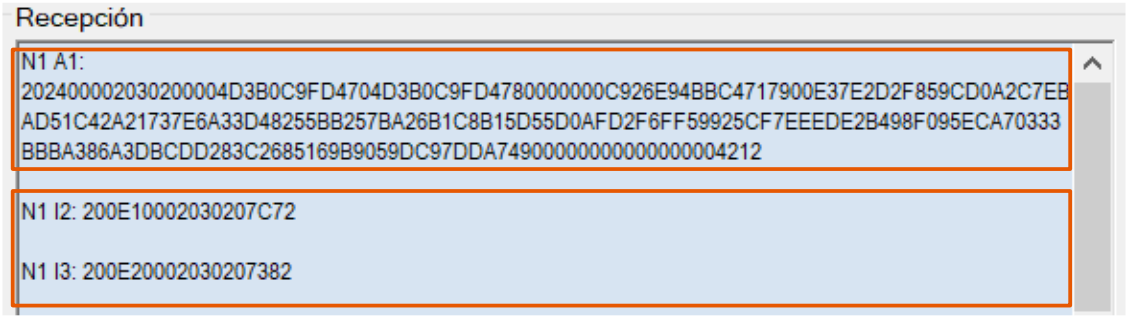


Figura 3.9. Tramas A1, I2, I3 recibidas por el *Hub*.

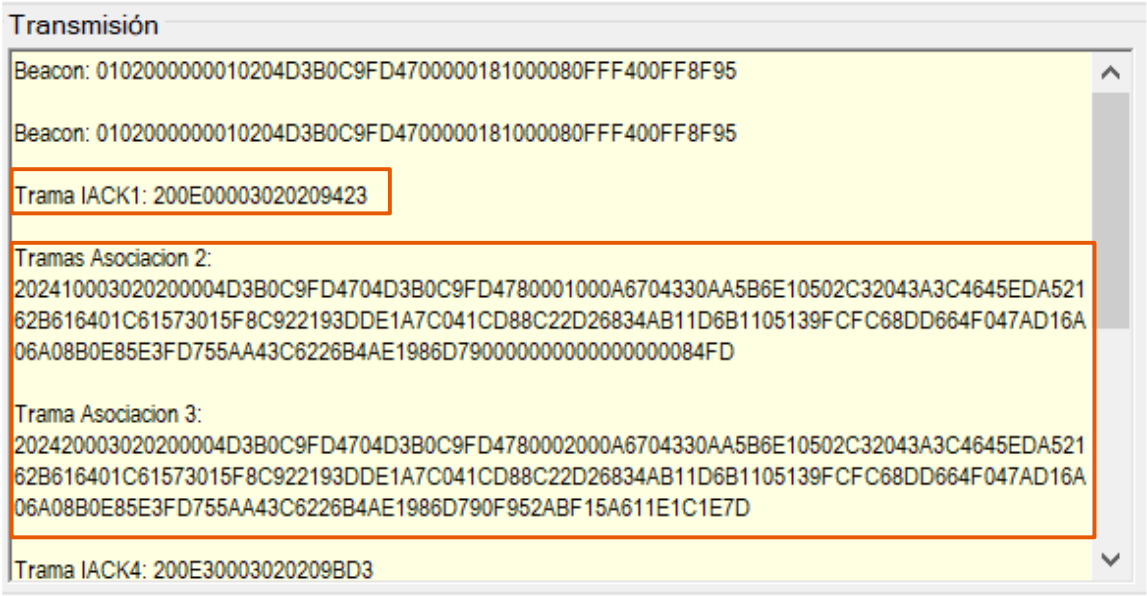


Figura 3.10. Tramas I1, A2, A3 enviadas por el *Hub*.

Figura 3.11. Información calculada al recibir las tramas A1 e I2 en el *Hub*.

Cuando el Nodo recibe la trama A2, envía un IACK-2 y calcula la clave Diffie-Hellman, llave temporal, MK_KMAC_3A y MK_KMAC_4A. Después al recibir la trama A3, envía el correspondiente IACK-3 y pasa a formar la trama A4 que será transmitida al *Hub*, recibiendo como respuesta a esta última un IACK-4. Las Figuras 3.12, 3.13 y 3.14 muestran el proceso descrito anteriormente el cual se desarrolla solo en el Nodo.

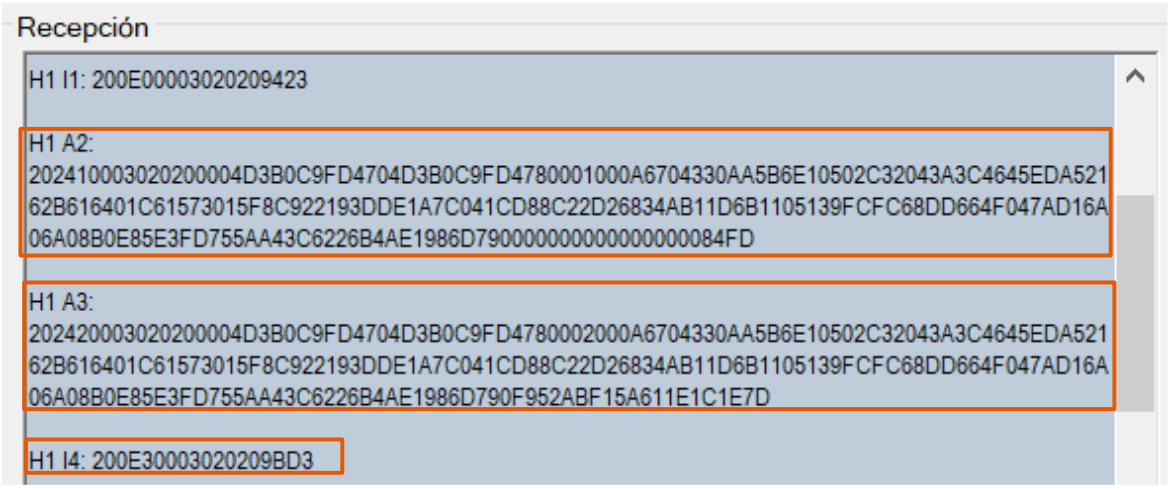


Figura 3.12. Recepción de las tramas A2, A3 e I4 en el Nodo

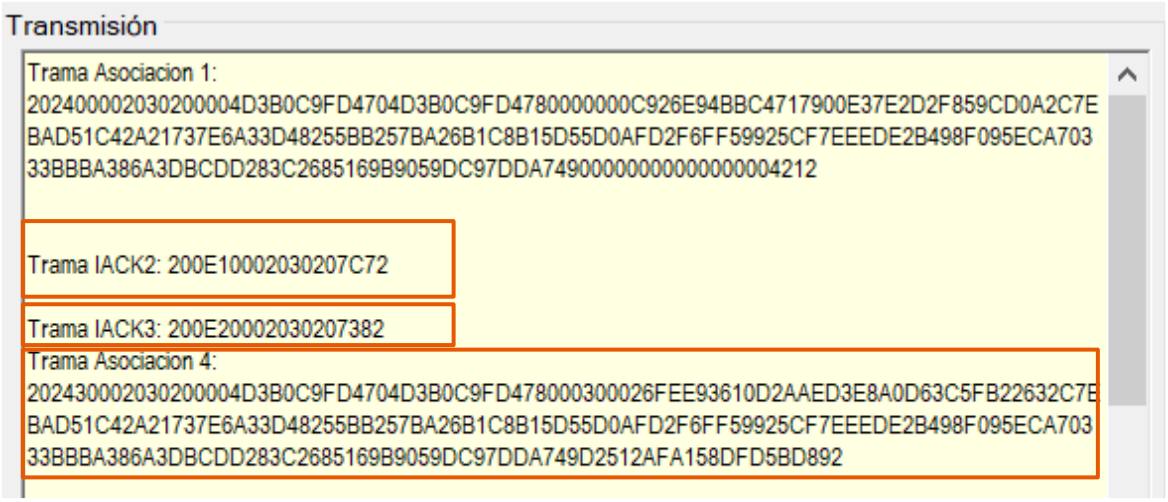


Figura 3.13. Transmisión de las tramas I2, I3 y A4 desde el Nodo.

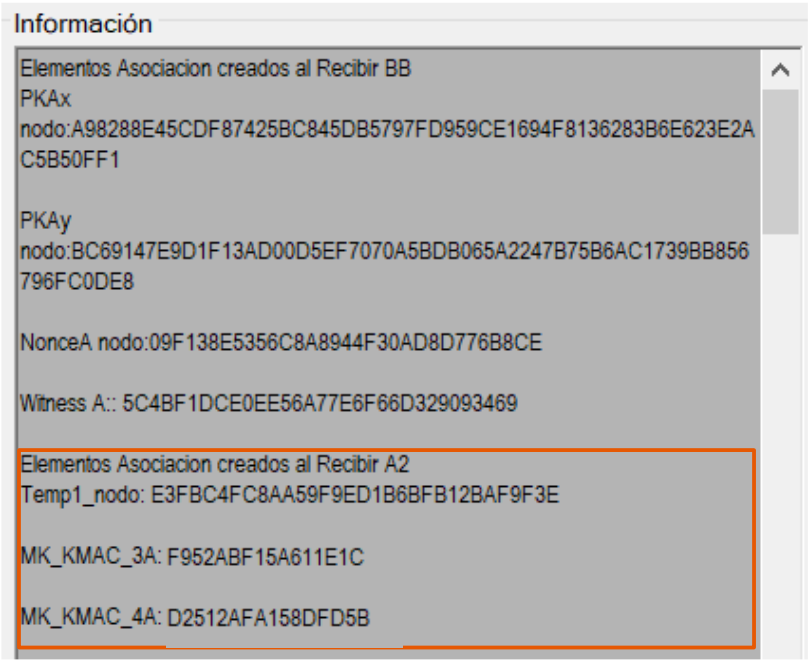


Figura 3.14. Cálculo de elementos en el Nodo.

Una vez que el Nodo ha receptado la tercera trama, se verifica que el MK_KMAC_3B y el MK_KMAC_3A coincidan para poder enviar la última trama de este proceso. Cuando el *Hub* recibe la cuarta trama envía el acuse, y después se realiza la misma verificación que en el Nodo, con los MK_KMAC correspondientes; si pasa la comprobación entonces se realiza el cálculo del *Display* en el *Hub*. De la misma manera si se ha comprobado la autenticidad de la trama tres entonces el Nodo calcula su *Display* al recibir la trama IACK-

4. Al final las dos entidades activan la MK, que debe ser la misma para las dos entidades. La Figura 3.15 muestran los valores del Display calculados tanto en el Nodo como en el *Hub*, que deben ser idénticos en ambos lados y la posterior activación de la MK en cada entidad.

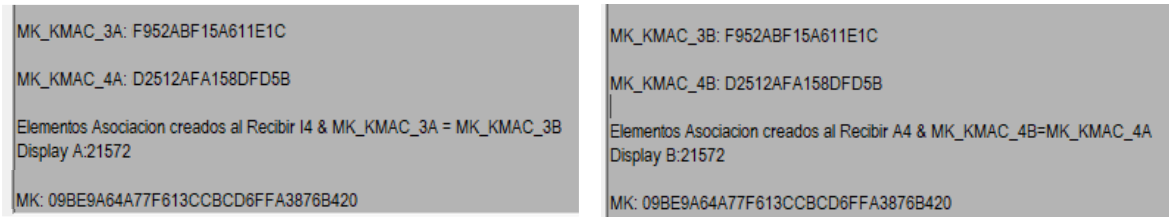


Figura 3.15. Cálculo del *Display* y activación de la MK en el Nodo/*Hub*.

Si en algunos de los procesos de verificación MK_KMAC de las entidades, los valores no corresponden, el proceso mostrará un mensaje de error y se cerrará la aplicación, obligando a reiniciar las soluciones de ambas entidades.

3.1.2. PROCESO DE CREACIÓN PTK

Una vez que se ha activado correctamente la MK en cada entidad, es posible pasar al proceso de Creación PTK. Esta segunda etapa de seguridad permite realizar una nueva autenticación de entidades. El proceso es más simple que el anterior ya que el número de tramas de intercambio disminuye, así como el cálculo de los elementos.

El Nodo es el encargado de enviar la primera trama hacia el *Hub*, para lo cual, primero calcula su *Nonce*. Una vez que el *Hub* recibe la trama, envía el IACK-5 para confirmar la llegada de la trama PTK1 y calcula su respectivo *Nonce*R, PTK, KCK, P, PTK_KMAC_2B y PTK_KMAC_3B. Inmediatamente de efectuados los cálculos, se envía la segunda trama con el PTK_KMAC_2B para la autenticación con el Nodo. Cuando el Nodo recibe la segunda trama, primero envía su correspondiente acuse de recibo, después calcula los elementos para la autenticación con el *Hub* como son: PTK, KCK, P, PTK_KMAC_2A y PTK_KMAC_3A. Una vez obtenido estos elementos que se encuentran relacionados entre sí, se verifica la identidad de la entidad que envía la trama, comparando los KMAC correspondientes. Con el éxito de esta verificación, se envía la tercera trama desde el Nodo para que se realice el mismo proceso de comprobación KMAC en el lado del *Hub*. Cuando se han verificado las identidades de las entidades entonces se activa la PTK en cada una.

La Figura 3.16 junto a la 3.17 muestran el proceso de intercambio de tramas entre las entidades Nodo y *Hub* para el cumplimiento del proceso en el párrafo anterior. Los colores

en cada entidad son correspondientes a los procesos que se dan a continuación de enviarse cada trama. Al igual que en el proceso de Asociación si alguna de las comprobaciones falla, se enviará un mensaje de error seguido de la finalización de la aplicación, invitando así a reanudar el proceso de las entidades, pero desde el proceso de Asociación.

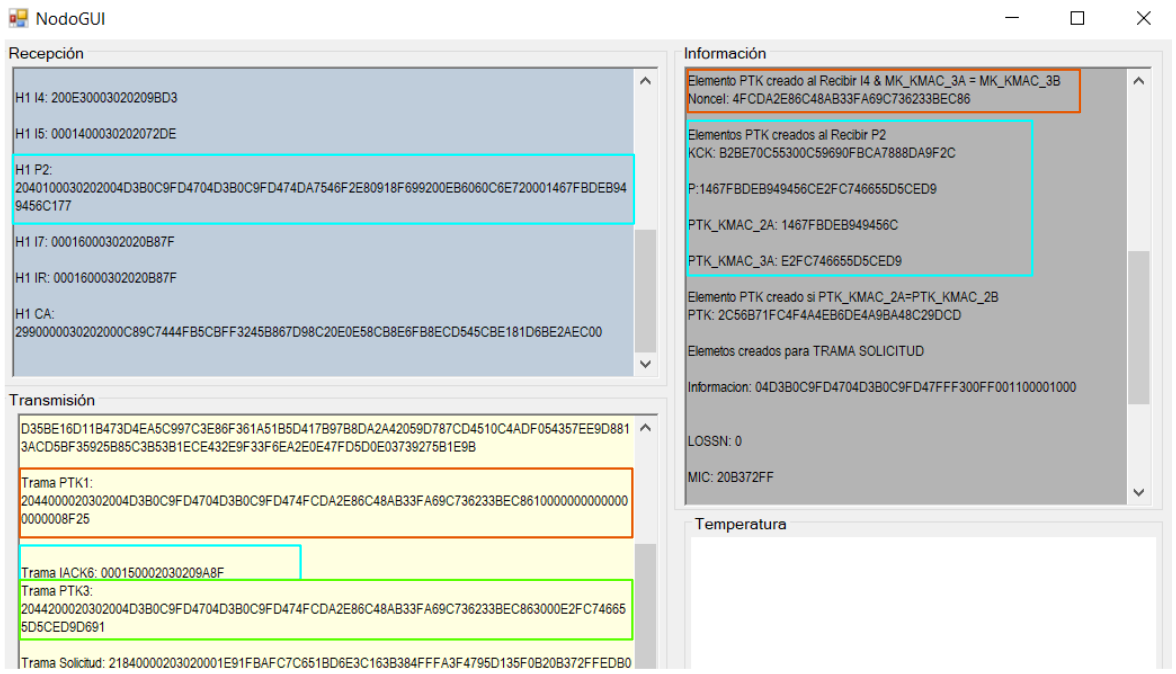


Figura 3.16. Proceso de Creación PTK en el Nodo.

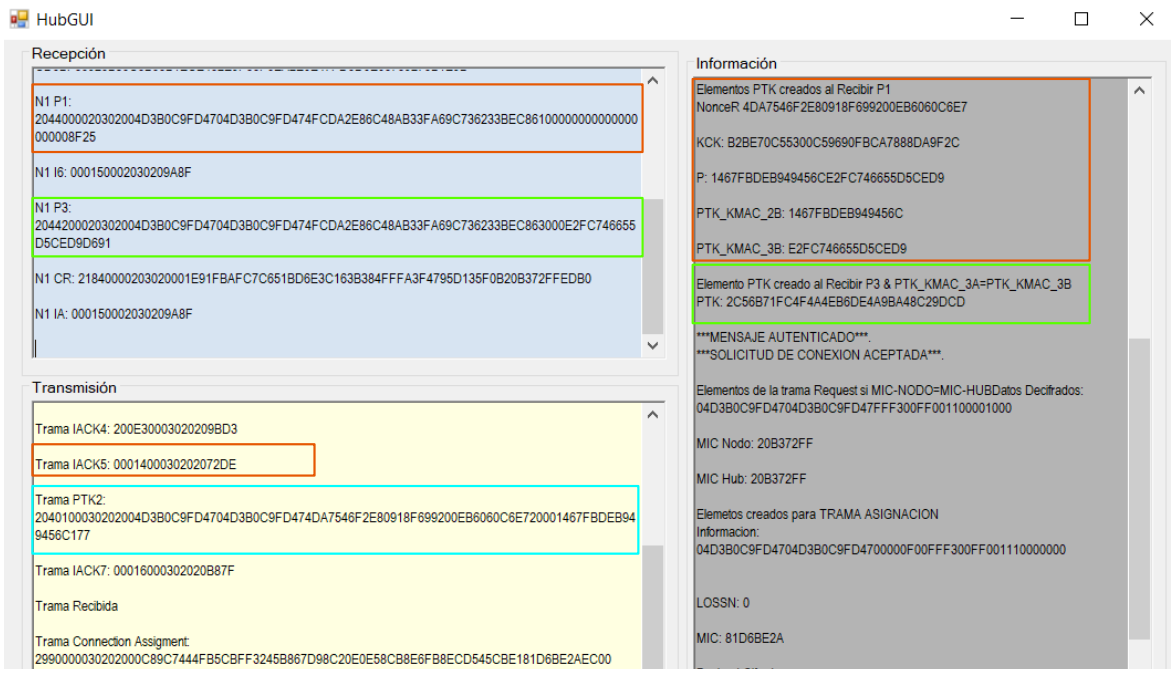


Figura 3.17. Proceso de Creación PTK en el *Hub*.

3.1.3. SEGURIDAD DEL MENSAJE

Las tramas que se intercambian entre las entidades autorizadas, una vez que se ha creado el PTK, deberán ser autenticadas y cifradas, porque se ha establecido un nivel 2 para asegurar la información. De acuerdo a lo establecido en el estándar IEEE 802.15.6 las tramas de Solicitud y Asignación de Conexión que se transmiten cuando las entidades se encuentran en el estado *Secured,* deben ser tramas aseguradas, debido a que se forman después que se ha creado el PTK, por lo tanto, deberán tener cifrado y autenticación como lo establece el modo de operación CCM.

Para la autenticación de la trama, se asigna un número de secuencia y se calcula la etiqueta MIC, después se cifra el *Payload* y por último se arma la trama Solicitud de Conexión para ser transmitida al *Hub.* Cuando el *Hub* recibe la trama, envía el respectivo acuse recibo IACK-R.

En el *Hub* se calcula la etiqueta MIC, en primer lugar, descifrando los datos, y se compara con la etiqueta enviada en la trama; si el resultado es positivo, es decir si ambas son iguales, entonces se procede a calcular la trama de Asignación. Como el envío es de *Hub* a Nodo entonces el *Hub* llevará su propio contador de número de secuencia porque así lo establece el estándar.

El procedimiento para la trama Asignación es igual que en la trama Solicitud, después de cifrar la información y junto con su correspondiente número de secuencia y etiqueta, se envía la trama al Nodo para que este último evalúe el elemento de autenticación con los mismos procedimientos que en el *Hub*. Si el intercambio de tramas de Solicitud y Asignación es exitoso, entonces las entidades pasan al estado *Connected,* en donde pueden realizar la transferencia de información relacionada al ser humano. En este estado las tramas de datos que se transmiten son aseguradas, obteniendo los beneficios de autenticación, privacidad, *data freshness*.

Las Figuras 3.18 y 3.19 muestran el proceso de transferencia de las tramas de Solicitud y Asignación, descrito anteriormente, que tiene lugar entre el Nodo y *Hub*. Si alguna de las verificaciones falla, entonces se despliega un mensaje de error y se solicita el cierre de la aplicación, obligando al usuario a reiniciar el proceso desde el proceso de Asociación de las entidades.

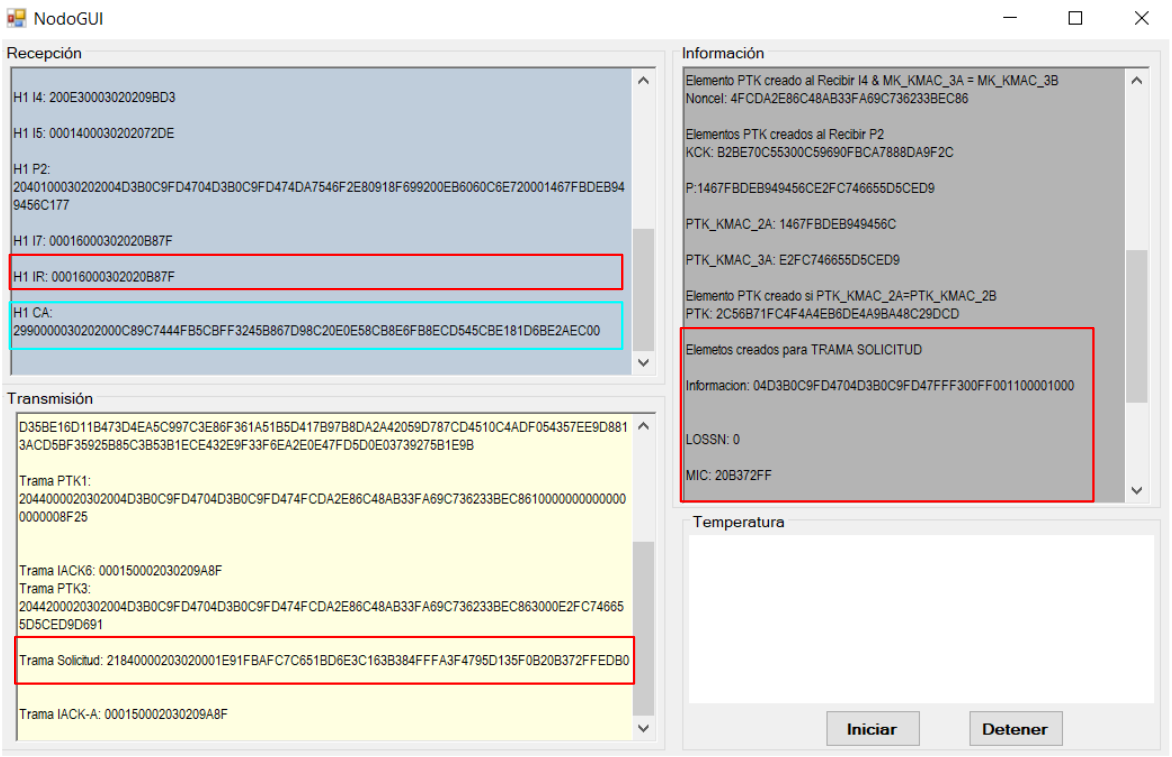


Figura 3.18. Transmisión de las tramas CR e IACK-A en el Nodo.

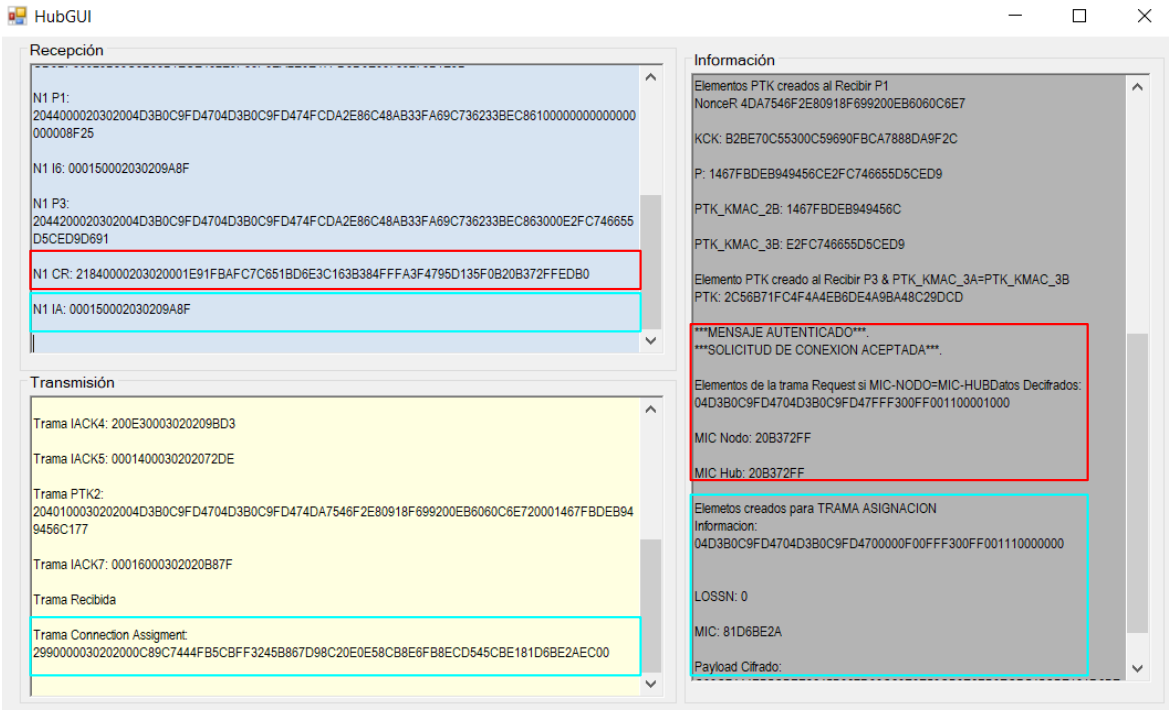


Figura 3.19. Transmisión de las tramas IACK-R y CA en el *Hub.*

En la Figura 3.20 se muestran en consola los bloques que se irán creando para la autenticación y cifrado del *Payload* de la trama, la creación de los bloques: *Nonce*, B0, Ctr se realiza tanto en el Nodo como el *Hub*. Además, se muestra la división en bloques de 32 hexadecimales de la información obtenida. En la Figura 3.20, a la izquierda se muestra la salida por consola del Nodo donde se indica los elementos creados: *Nonce*, Ctr y división en bloques de la información B, para la creación de la etiqueta MIC. A la derecha, en la misma figura, se muestra la creación de los mismos elementos para calcular la MIC, dividiendo el *Payload* cifrado en bloques de 32 hexadecimales, para después descifrarlos y así poder calcular la etiqueta, para en lo posterior, compararla con la envida por el Nodo. Por lo tanto, los elementos que crea el *Hub* deberán ser los mismos que el Nodo, para que el resultado arrojado sea el mismo.

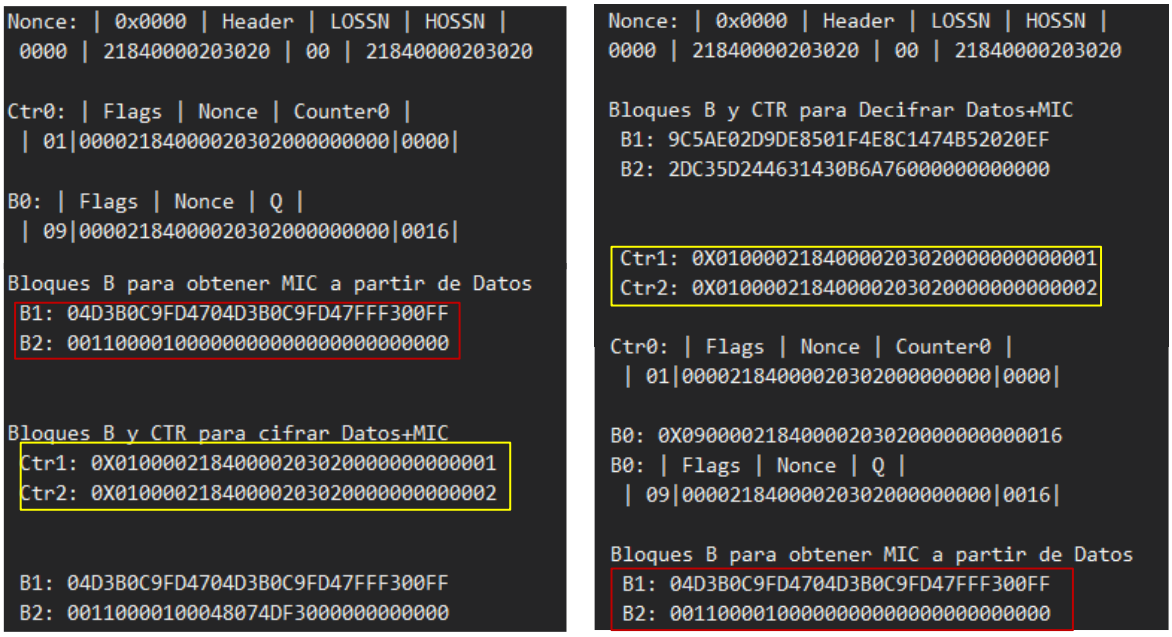


Figura 3.20. Bloques para Cifrado y Autenticación. (a) Nodo. (b) Hub.

3.2. PRUEBAS DE CONCEPTO - PARADIGMA DE SEGURIDAD

La presente sección se enfocará en mostrar la seguridad que puede proveer las etapas dos y tres del paradigma de seguridad, las cuales son: Creación PTK y Seguridad del Mensaje, mediante la ejecución de pruebas de concepto a cada una de las fases mencionadas. Para la realización de estas pruebas se definirán los escenarios donde se podrían encontrar fallas de seguridad que afecten el buen desempeño del paradigma;

además también se establecerán las respectivas condiciones para cada escenario propuesto. La red de comunicación utilizada para los fines planteados se indica en la Figura 3.2.

3.2.1. ESCENARIO DE PRUEBA 1

Este escenario prueba la seguridad del proceso de Creación PTK, el cual se basa en la utilización de la herramienta de cifrado CMAC, que permite detectar modificaciones tanto intencionales como accidentales y a su vez permite autenticación. En el contexto del proceso dos del paradigma, CMAC se utiliza para asegurar la creación del PTK, dando garantía que los elementos creados en este proceso sean únicos y por ende ningún intruso pueda reproducir la clave o los elementos que se crean en las entidades autorizadas. Para este escenario se establecen algunos requisitos que se mencionan a continuación:

- Nodo, *Hub* y Equipo A deberán tener una conexión al Servidor, y el Equipo A empezará la escucha antes que inicie el proceso de Creación PTK.
- El Equipo A tendrá las herramientas necesarias para realizar la prueba y esto incluye el código.
- Se asume que el proceso de Asociación se ha realizado con éxito entre las entidades autorizadas, y que, por lo tanto, el MK creado es seguro y ningún otro equipo lo posee.

Las entidades Nodo y *Hub* empiezan el proceso de Asociación hasta completar con éxito la creación de la MK, mientras tanto el Equipo A se mantiene escuchando la comunicación.

Cuando el proceso para crear la PTK inicia, el Nodo envía la primera trama PTK y tanto el *Hub* como el Equipo A la reciben y en ese instante ambas entidades calculan los elementos necesarios para formar la segunda trama PTK. Los elementos calculados en cada entidad son: KCK, P y PTK-KMAC-2B, una vez creados se estructura y envía la segunda trama, desde el *Hub* y el Equipo A. El *Hub* crea los elementos con la llave MK real, mientras que el Equipo A, al no poseerla lo hace con una MK falsa.

En este escenario se producen dos condiciones, la primera cuando la trama P2 del *Hub* llega al Nodo antes que la P2 del Equipo A y la segunda cuando la trama P2 del Equipo A llega al Nodo antes que la trama P2 del *Hub*.

En el primer punto, al llegar la trama P2 al Nodo, se compara el PTK_KMAC_2B enviado por el *Hub* con el PTK_KMAC_2A calculado en el Nodo, lo cual da como resultado un éxito, ya que estos valores se han calculado con la clave que poseen las dos entidades

autorizadas. Al llegar la trama del Equipo A en segundo lugar, ya no se procesa la trama y se descarta.

La Figura 3.21 muestra la interfaz del Nodo, en la sección de Recepción se puede diferenciar la llegada de las tramas P2 por parte del *Hub* y el Equipo A distinguiéndose por medio de los identificadores de entidad. Como se mencionó anteriormente la trama por parte del *Hub* llega primero y se calcula con éxito la clave PTK, como se indica en la sección de Información. La trama transmitida por parte del Equipo A se descarta como se aprecia al final en la sección Información. Ya que los elementos de comparación son exitosos, el *Hub* envía la tercera trama para poder activar la PTK en las entidades autorizadas.

Al llegar en segundo lugar la trama del Equipo A, los procesos no se interrumpen y ninguna entidad autorizada debe reiniciar sus actividades, lo que permite continuar con el proceso dos y tres y así concluir el paradigma de seguridad.

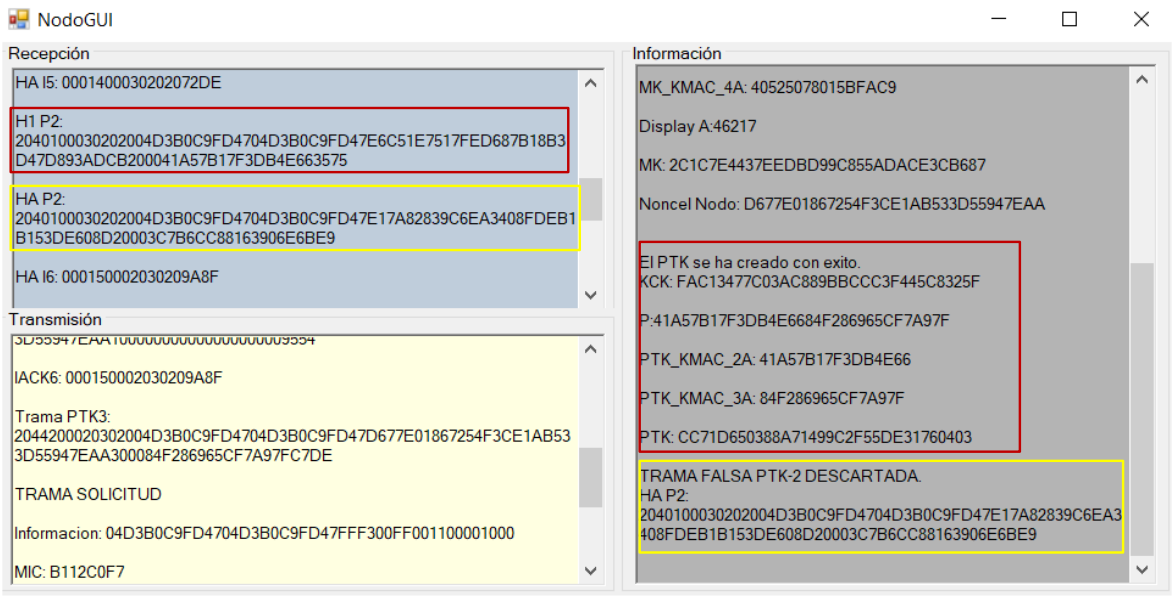


Figura 3.21. Recepción de la trama P2 del Hub en primer lugar.

El segundo punto trata sobre la llegada de la trama P2 del Equipo A antes que la del *Hub*, el momento de la autenticación de la trama P2 enviada por parte del Equipo A, falla por no poseer la MK para crear los elementos que generan el PTK_KMAC. En la Figura 3.21 se muestra en la sección de Recepción la llegada de la trama P2 del Equipo A, mientras que en la de Información se indica que el proceso ha fallado debido a que las credenciales de comprobación no son las correctas.

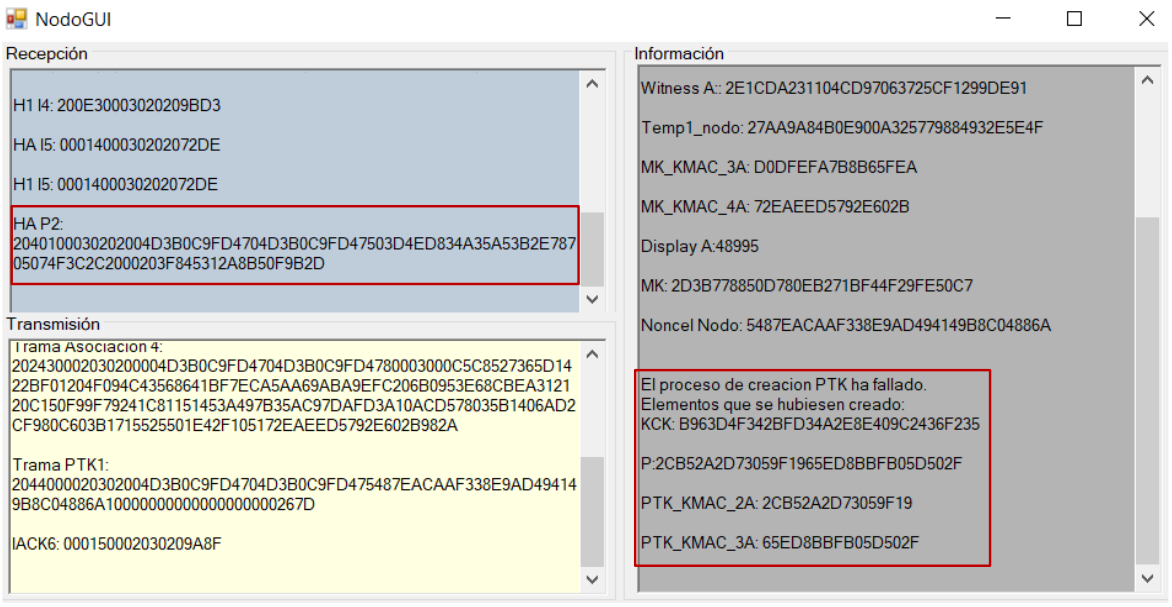


Figura 3.22. Recepción de la trama P2 del Equipo A en primer lugar.

El Equipo A además de enviar la trama P2 hacia el Nodo también envía la trama P3, que calcula con la MK falsa, hacia el *Hub*. Al recibirla el *Hub* realiza las comprobaciones del PTK_KMAC y el proceso se aborta. Lo indicado aquí se muestra en la Figura 3.23, donde se aprecia la llegada de la trama P3 por parte del Equipo A en la sección Recepción y un mensaje de falla en el proceso.

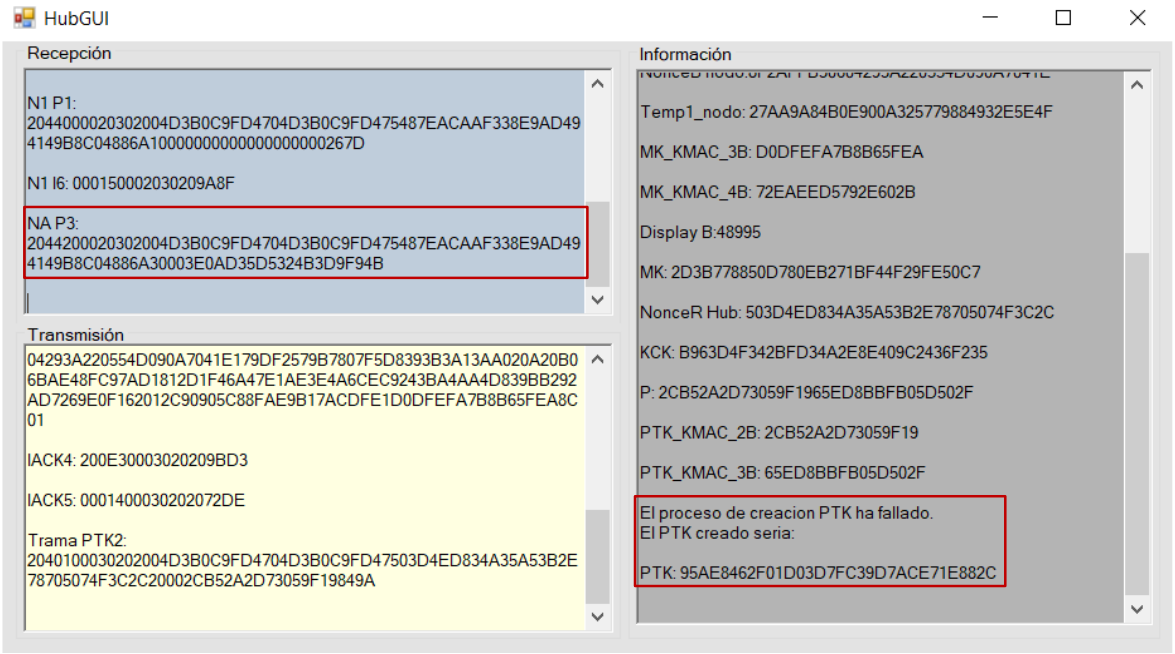


Figura 3.23. Recepción de la trama P3 enviada por Equipo A.

En la Figura 3.24 se observa las tramas P2 y P3 que se han transmitido desde el Equipo A hacia el Nodo y el *Hub* respectivamente. Si se observa, estas tramas son las que llegan en las Figuras 3.22 y 3.23 respectivamente.

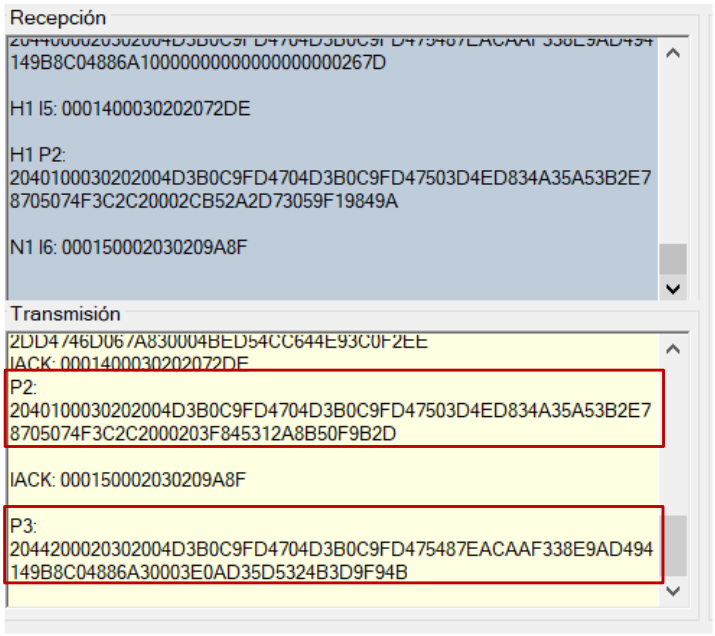


Figura 3.24. Transmisión de las tramas P2 y P3 desde el Equipo A.

Al producirse este fallo, el proceso deja de llevarse a cabo y se aborta debido a que las comprobaciones no se han ejecutado con éxito. Como se observa, cuando el Equipo A intenta realizar una falsificación de las tramas PTK para establecer una comunicación entre las entidades autorizadas, se encuentra con una barrera de cifrado que no le permite ejecutar su objetivo; pero en contraste, esto obliga a que el proceso se detenga y que deba reiniciarse, volviéndose a encontrar con una entidad no autorizada que se encuentre escuchando el intercambio de tramas y retornando al mismo problema que se ha mencionado.

La cantidad de veces en que el Equipo A pueda impedir que el proceso de Creación PTK se ejecute con éxito, depende de que tan rápido la trama PTK falsa llegue a cualquiera de las entidades. Para una serie de intentos realizados se han obtenidos los siguientes resultados.

Durante el proceso, para crear la clave PTK, la trama falsa logró ingresar antes que la verdadera un máximo de tres veces consecutivas de acuerdo a la Tabla 3.1, esto quiere decir que después de esto, las entidades lograron completar con éxito la creación de la

clave. En la Tabla 3.1 se muestra la cantidad de veces que el Equipo A pudo denegar los servicios de las entidades. El éxito de una intrusión se representa por la letra E en la primera fila, mientras que, el fracaso con F en la segunda.

Tabla 3.1. Resultados de las pruebas del escenario 1.

	Intentos																											
E	X	X	X				X	X	X				X	X	X			X	X			X			X	X	X	
F				X	X	X				X	X	X				X	X			X	X		X	X				X

Los resultados de la Tabla 3.1 indican que se puede dar una denegación de servicios, sin embargo, no se puede mantener este resultado por mucho tiempo, ni durante muchos intentos de ingreso, ya que una vez que la trama real ha ingresado con éxito, la falsa se descarta y el proceso sigue el curso normal de autenticación hasta concluirlo.

3.2.2. ESCENARIO DE PRUEBA 2

El proceso de Seguridad del Mensaje es el tercero y último del paradigma y provee varios servicios de seguridad para las tramas Solicitud y Asignación de la Conexión, así como a las tramas de datos cuando se ha pasado al estado *Connected*. En este escenario se prueba la autenticación, basado en la creación de etiquetas para cada trama transmitida. La etiqueta MIC correspondiente a cada trama es el resultado de un proceso que incluye varios elementos y los propios datos a ser transmitidos, junto con un proceso de cifrado que ahonda más en la complejidad de poder falsificarla por terceros.

En este escenario se establecen algunos requisitos que se mencionan a continuación:

- Nodo, *Hub* y Equipo A deberán tener una conexión al Servidor, y el Equipo A se mantiene escuchando la comunicación constantemente.
- El Equipo A tendrá las herramientas necesarias para realizar la prueba.
- Se asume que el proceso de Asociación se ha realizado con éxito entre las entidades autorizadas, y que, por lo tanto, la MK creada es segura y ningún otro equipo lo posee al igual que la clave PTK.

En el presente escenario se capturan las tramas enviadas desde el Nodo hacia el *Hub* para modificar un hexadecimal de cada trama y retransmitirlas hacia el *Hub*. Considerando que el PTK no ha podido ser obtenido, se realizan las modificaciones a las tramas.

El tratamiento que se da a las tramas que han sido enviadas por el Nodo y el Equipo A es el mismo, el cual se menciona a continuación: el *Hub* recibe una trama de datos y calcula

su propio MIC a partir del *Payload* y el número de secuencia que forman parte de la trama recibida, después compara estos valores y toma una decisión de autenticación o descarte.

En la Figura 3.25 se muestra en la sección de Información los elementos calculados para la identificación de las tramas, que, en caso de ser exitoso, aparece al inicio con un mensaje *MENSAJE AUTENTICADO* y de lo contrario como *FALLA DE AUTENTICACION.* En cada caso se calculan los elementos para apreciar los valores de etiqueta MIC. Cuando la trama es auténtica y procede de la entidad autorizada que contiene las llaves en común con el *Hub,* las etiquetas MIC del Nodo y *Hub* son iguales; mientras que cuando se ha cambiado uno solo de los valores de la misma trama, al tratar de calcular la MIC en el *Hub* éste da como resultado un valor totalmente diferente, por lo que la trama es descartada.

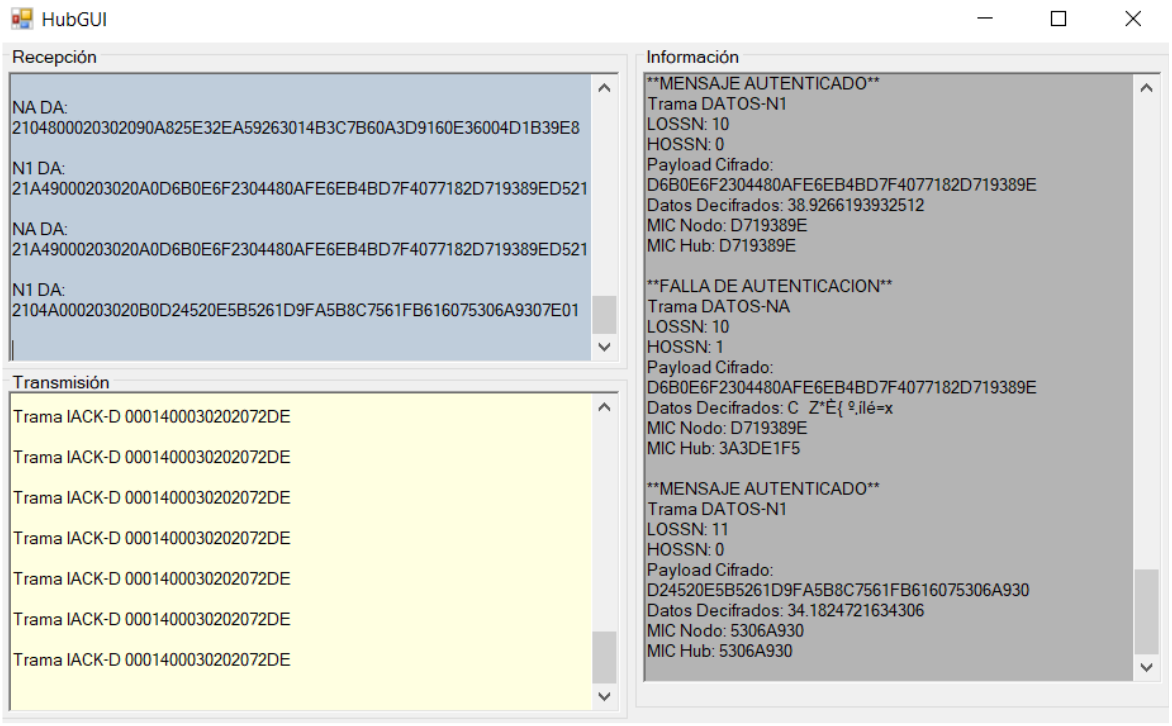


Figura 3.25. Recepción de tramas auténticas y modificadas en el *Hub*.

Para el propósito de autenticación en este escenario, se ha enviado diferentes cantidades de tramas, auténticas como modificadas al *Hub*, los resultados se indican en la Tabla 3.2. El número de secuencia de bajo orden LOSSN y el número de secuencia de alto orden HOSSN dan el total de la cantidad de tramas recibidas en el *Hub*, a través de la Ecuación 3.1, estos valores son tomados de la GUI en el *HUB*, de la última trama con *MENSAJE AUTENTICADO*.

Los resultados de las distintas muestras de tramas, arrojan la seguridad esperada, un total de cero tramas modificadas que no han podido ser autenticadas. Esto se explica porque el cálculo de la MIC requiere del *Nonce*, B0 y los datos mismos, al modificar una parte de ellos el proceso para calcular la MIC arroja un resultado que no concuerda con el enviado en la trama y al realizar el proceso de comparación el resultado no es el mismo y por lo tanto la trama debe ser descartada. En un entorno real, la información de la trama se ve afectada por el canal de transmisión, puede suceder que un bit cambie de manera accidental, y por lo tanto la información se modifica y sucede lo mismo que con una modificación intencional evitando que información errónea en este caso se procese.

Tabla 3.2. Resultados del escenario 2.

LOSSN	HOSSN	Total Tramas Recibidas	Éxito Intrusiones
5	5	1285	0
246	7	2038	0
16	17	4368	0
56	40	10296	0

$$TotalTramas = LOSSN + 256 * HOSSN \quad (3.1)$$

El último de los resultados obtenidos de acuerdo a la Tabla 3.2 se muestra en la Figura 3.26, donde se observa la llegada de las tramas de datos en la ventana Recepción, del Nodo y el Equipo A, reconocidas por medio de los identificadores N1 y NA. Mientras tanto, en la ventana Información se indica si estas mismas tramas han sido autenticadas.

3.2.3. ESCENARIO DE PRUEBA 3

Una característica importante de la Seguridad del Mensaje es la *data freshness*, ya que ayuda a que el mensaje recibido sea el actual. En este escenario de comunicación se evaluará este servicio, para lo cual se establecen algunos requisitos que se mencionan a continuación:

- Nodo, *Hub* y Equipo A deberán tener una conexión al Servidor, el Equipo A se mantiene escuchando la comunicación permanentemente.
- Se asume que el proceso de Asociación se ha realizado con éxito entre las entidades autorizadas, y que, por lo tanto, el MK creado es seguro y ningún otro equipo lo posee al igual que la clave PTK.

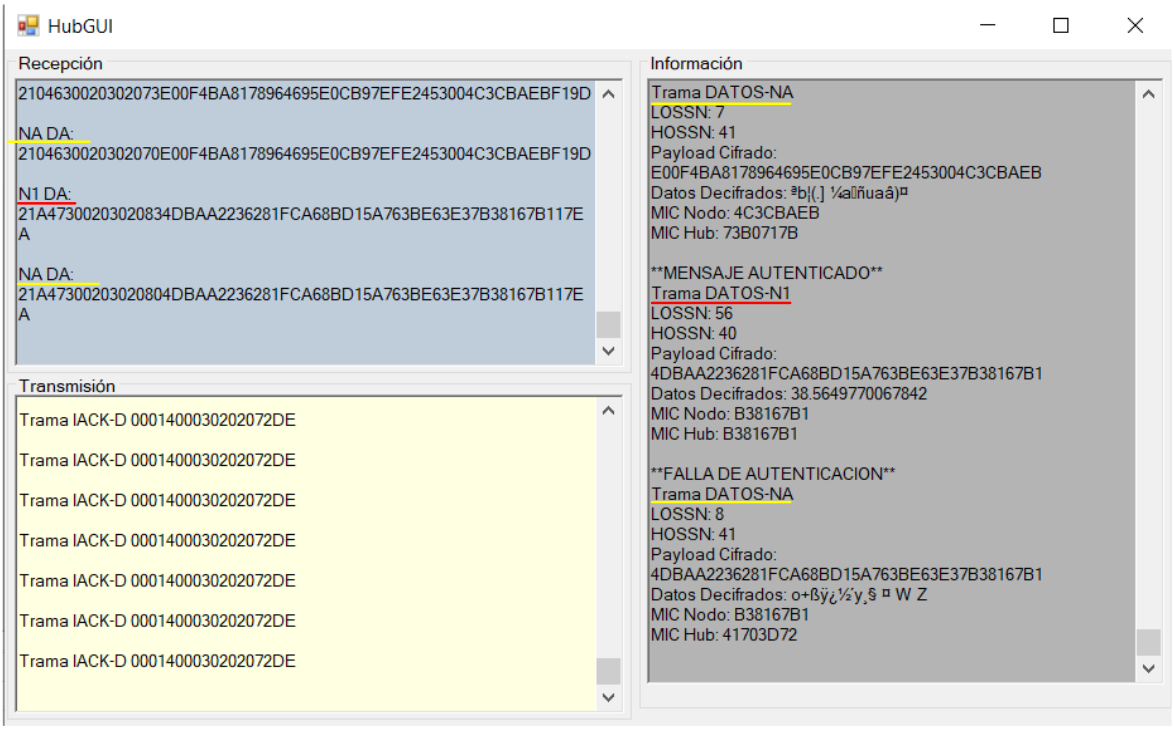


Figura 3.26. Resultados de las pruebas en el Hub.

En este escenario se envían tramas desde el Nodo al *Hub*, las cuales son capturadas por el Equipo A y almacenadas para posteriormente reenviarlas sin ninguna modificación. Todas las tramas enviadas por el Equipo A son de la misma colección, es decir que están aseguradas con el mismo PTK. Para probar la *data freshness*, se detiene el envío de tramas desde el Nodo, como el Equipo A ha guardado todas las tramas enviadas por el Nodo éste las reenvía como una réplica exacta al *Hub* para tratar de que las autentique y poder confundir a la entidad evaluadora. Se han enviado diferentes números de tramas réplica y se presentan los resultados en la Tabla 3.3.

Tabla 3.3. Resultados del escenario 3.

LOSSN	HOSSN	Tramas Nodo	Tramas Equipo A	Tramas descartadas
10	1	266	266	266
127	4	1151	1151	1151
165	8	2213	2213	2213
15	16	4111	4111	4111
96	34	8800	8000	8000

El número de tramas enviado del Equipo A es igual al del Nodo debido a que el primero empieza a almacenar las tramas de datos apenas éstas llegan. Cuando las tramas del

Equipo A se envían a pesar de ser tramas verdaderas éstas fallan en su intento de autenticación, esto gracias al número de secuencia de bajo orden enviado por la trama y al de alto orden obtenido en la entidad. Los números de secuencia como se ha indicado antes, ayudan a mantener el "*freshness*" del mensaje, esto es que no permite el ingreso de tramas antiguas. La Figura 3.27 muestra el resultado de las pruebas realizadas cuando se envían las tramas duplicadas.

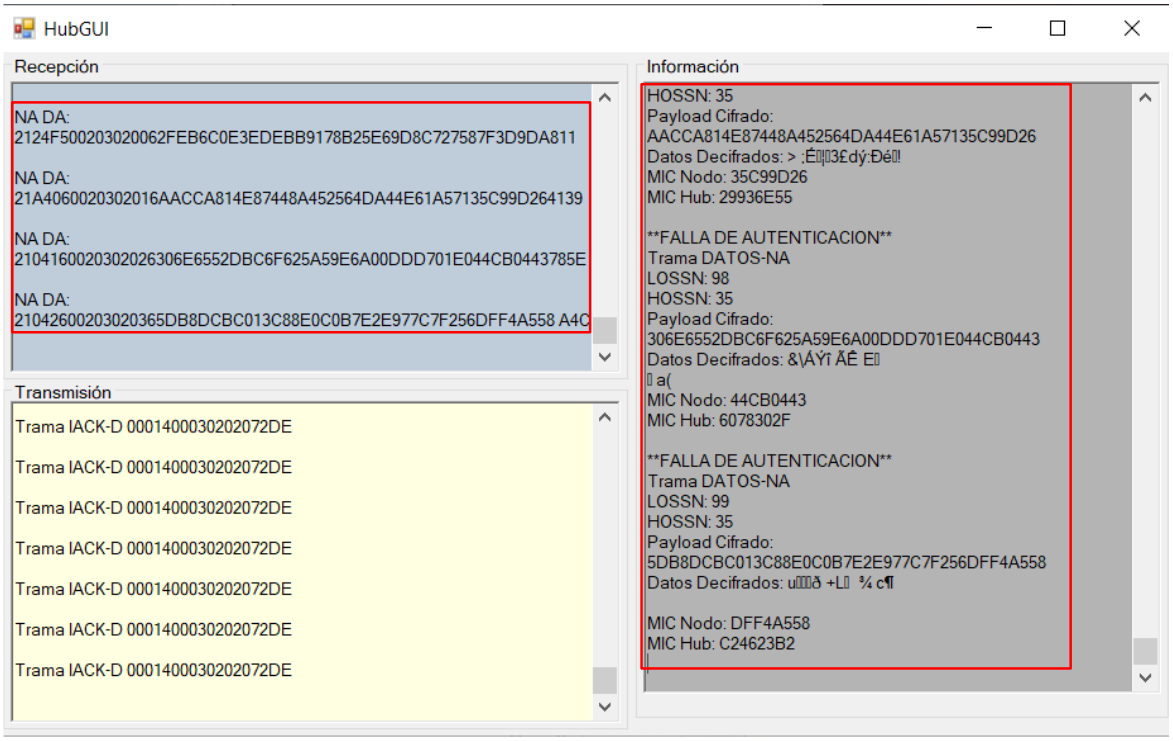


Figura 3.27. Resultados de las pruebas.

En la Tabla 3.4 se presentan los casos, mediante un ejemplo, en los que las tramas replicadas pueden encontrarse respecto de una actual. La nomenclatura utilizada se menciona a continuación:

- Número de Secuencia de bajo orden del Nodo (L_N).
- Número de Secuencia de alto orden del Nodo (H_N).
- Número de Secuencia de bajo orden del *Hub* (L_H).
- Número de Secuencia de alto orden del *Hub* (H_H).

Las tramas replicadas no pueden ser autenticadas, debido a que en el destino se lleva la cuenta de los dos números de secuencia, cuando se forma el *Nonce* lo hace con ambos números de secuencia, el de bajo orden de la fuente y el de alto orden del destino, por lo

que el resultado dará una secuencia distinta de la esperada y por lo tanto la trama debe ser descartada y se deben mantener los valores de los números de secuencia del destino para cuando una trama actual llegue.

Tabla 3.4. Casos de llegada de tramas antiguas.

Caso	Valores en la fuente	Valores en el destino	Resultado
$L_N < L_H$	L_N =129 H_N=6	L_H =130 H_H =7	No se autentica
$L_N = L_H$	L_N =130 H_N=6	L_H =130 H_H =7	No se autentica
$L_N > L_H$	L_N =131 H_N=6	L_H =130 H_H =7	No se autentica

3.2.4. ESCENARIO DE PRUEBA 4

En este escenario se evaluará la privacidad, que tiene como objetivo impedir que se pueda revelar el mensaje ante personas que no poseen las claves necesarias para descifrarlo. Al no poseer las claves, se utilizan mecanismos para tratar de descifrarlos buscando debilidades en los algoritmos que puedan ser utilizadas para este fin. Un ataque puede ocurrir cuando el valor de un *Nonce* se duplica, es decir que para una misma clave de cifrado se encriptan dos mensajes con el mismo *Nonce*.

Dado un par de mensajes cifrados como lo muestran la Ecuación 3.2 y 3.3, si se aplica una operación XOR a ambos mensajes cifrados, se tiene como resultado la Ecuación 3.4. El resultado de la Ecuación 3.4 se debe a que la función AES siempre produce la misma salida cuando se tienen las mismas entradas.

$$C_1 = M_1 \oplus AES(N, K) \tag{3.2}$$

$$C_2 = M_2 \oplus AES(N, K) \tag{3.3}$$

$$C_1 \oplus C_2 = M_1 \oplus M_2 \tag{3.4}$$

Donde:

C_1, C_2: resultado de aplicar una función de cifrado a un texto plano.
M_1, M_2: texto plano.
AES: función de cifrado AES.
N: *Nonce*.
K: clave para el cifrado.

Como ya se mencionó, esta debilidad ocurre cuando un *Nonce* se repite, de acuerdo al modo de operación CCM utilizado en el estándar IEEE 802.15.6, podría ocurrir cuando se ha superado el número máximo de valores permitidos, es decir después de 2^{32}. Como se observa éste es un número alto de *Nonce*s, donde una persona no autorizada debería estar capturando tramas hasta que llegue a este valor. El presente escenario pretende evaluar la privacidad tomando distintos conjuntos de tramas reales para efectuar la operación XOR y establecer si es factible obtener el texto sin cifrar de esta forma. A continuación, se establecen los requisitos para este escenario:

- Nodo, *Hub* y Equipo A deberán tener una conexión al Servidor.
- El Equipo A tendrá las herramientas necesarias para realizar la prueba y esto incluye el código para la captura, almacenaje.
- Se asume que el proceso de Asociación se ha realizado con éxito entre las entidades autorizadas, y que, por lo tanto, el MK creado es seguro y ningún otro equipo lo posee.

Cuando las entidades Nodo y *Hub* han establecido una clave PTK, el Equipo A puede iniciar la escucha una vez que se inicia la transferencia de tramas el quipo las captura y las guarda para poder realizar la operación XOR.

La captura de tramas se ha ido variando para analizar el tiempo necesario que toma realizar las operaciones. Las tablas 3.5, 3.6 y 3.7 indican el número de muestras tomadas y las claves que se utilizaron para cada una. Además, se incluye el tiempo utilizado para las diferentes muestras. En la primera muestra se toma un tiempo de 30 segundos para aplicar la operación XOR a todos los mensajes capturados. En la segunda muestra el tiempo de ejecución aumenta considerablemente respecto a la primera al igual que la tercera muestra respecto de las dos anteriores. Esto se debe a que las operaciones deben realizarse entre todas las tramas para esperar un resultado favorable.

Tabla 3.5. Tabla de Muestra 1.

	Muestra 1
Total tramas capturadas	95
Total tramas transmitidas	107
MK	888BDF4C09C8AFF105648C50BD8E085A
PTK	AC812A982CE63DF9C788F714433041DD
Tiempo ejecución	30 segundos

Tabla 3.6. Tabla de Muestra 2.

	Muestra 2
Total tramas capturadas	515
Total tramas transmitidas	515
MK	1A8852E580EDCF982C22BDA5D69E60F9
PTK	6CCAC4459B366DAB33308D0D8E9FB6F6
Tiempo ejecución	24 min 50 seg

Tabla 3.7. Tabla de Muestra 3.

	Muestra 3
Total tramas capturadas	649
Total tramas transmitidas	649
MK	1A8852E580EDCF982C22BDA5D69E60F9
PTK	6CCAC4459B366DAB33308D0D8E9FB6F6
Tiempo ejecución	56 min 40 seg

En ninguno de los casos se pudo revelar el mensaje y solo se obtuvo una secuencia de símbolos sin sentido, lo cual lleva a considerar que el objetivo de seguridad mediante cifrado cumple su finalidad, siempre y cuando no se reutilice el valor del *Nonce* bajo una misma clave PTK.

Por lo tanto, el uso de una misma clave PTK debe estar condicionada a la cantidad de *Nonce* que se puedan producir sin repetirse para evitar la revelación de la Información. Por otra parte, se tiene que el esfuerzo empleado para poder obtener un solo dato después de un tiempo considerable llevaría a cualquier atacante a desistir de este mecanismo y buscar otros que si produzcan un resultado más satisfactorio. Es así que también se puede efectuar un ataque de pre-cómputo, el cual consiste de calcular previamente valores para después compararlos con los de la ejecución hasta obtener un resultado favorable.

4. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

- Las fases dos y tres en el paradigma de seguridad deben gran parte de su seguridad a la primera etapa que corresponde a la Asociación, donde de ser el caso, si existe vulneraciones, no se puede contener o identificar la presencia del atacante, ni en el proceso de Creación PTK y tampoco en el de Seguridad del Mensaje. Por lo tanto, en caso de Asociación de un atacante, los procesos que siguen se encuentran comprometidos, por lo que, los datos que se transmitan serán falsos y esto seguirá mientras exista una conexión.

- El proceso de Creación PTK supone una debilidad que puede ser mejor explotada por un atacante, esperando realizar una negación de servicios para las entidades autorizadas, que conlleve un mayor número de intrusiones, aumentado el tiempo de indisponibilidad, hasta que las entidades decidan reiniciar el proceso desde el estado *Orphan.*

- En las pruebas realizadas para el proceso de Creación PTK, es verdad que el Equipo A no puede ingresar al proceso si este no tiene la clave PTK, pero también es cierto que puede negar este servicio. Esto debido a que, no se establecen medidas dentro del Estándar IEEE 802.15.6 que indiquen las acciones a tomar cuando una trama con un PTK_KMAC falso llegue a una entidad.

- La longitud del *Nonce* permite asegurar un número determinado de mensajes, específicamente 2^{32}, lo cual implica que después de este valor habrá que reiniciar todos los procesos, debido a que la duplicación del *Nonce* puede ser una debilidad explotada por terceros para revelar el mensaje, por lo cual es necesario evitar dicha inseguridad, comenzando nuevamente con los procesos del paradigma, obteniendo así una nueva PTK, para asegurar los mensajes.

- Con lo expuesto en el párrafo anterior, después de un cierto número de mensajes se puede llegar a obtener la información deseada, pero esto requiere de tiempo, en primer lugar, para capturar los mensajes y mucho más para revelar el mensaje. Así lo demuestra las pruebas de privacidad realizadas, en donde se indican que el tiempo tomado para efectuar el ataque, con diferente número de tramas, aumenta considerablemente conforme este último factor asciende. Además, el costo en

tiempo para obtener un éxito, con recursos limitados, no justifica el resultado que se pretende obtener.

- El proceso de descifrado implementado considera los distintos casos de números de secuencia que pueden arribar, siendo así que las tramas de Datos provenientes de una entidad autorizada se procesen correctamente, siempre que la trama sea actual y no una trama antigua, asegurando así el servicio de *Data Freshness.*
- La combinación de los algoritmos CMAC y AES permiten crear una secuencia fuerte de bits a partir de una clave secreta, como se observa en el proceso de Creación PTK, donde gracias a estos algoritmos la reproducción de los elementos de seguridad es imposible de realizarla en tiempo real.

4.2. RECOMENDACIONES

- El presente Proyecto Técnico abarca una comunicación *unicast,* donde se ilustra el proceso de comunicación en este ámbito, se puede completar el trabajo con la realización de un entorno *multicast*, el cual requiere de otro protocolo distinto para su realización.
- Se recomienda realizar la implementación del proceso de disociación para verificar la seguridad de este proceso en los diferentes estados de una comunicación segura. Además, también se puede implementar la desconexión a través de tramas inseguras como lo indica el estándar IEEE 802.15.6, para establecer las diferencias existentes entre disociación y desconexión y el papel en el ámbito de la seguridad del estándar.
- Se recomienda realizar la implementación de los niveles de seguridad 0 y 1, inferiores al tratado en este trabajo, para exponer las ventajas, desventajas de los mismos con respecto al nivel de seguridad 2 y en que aplicaciones sería recomendable utilizarlos.
- Las pruebas realizadas se han desarrollado en un entorno cableado, por lo que se debería cambiar las condiciones de red a una red inalámbrica que permita asemejar las condiciones sobre las cuales una WBAN trabaja.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] «Confidentiality and Privacy of Personal Data», *The National Academies Press*, 1994. [En línea]. Disponible en: https://www.nap.edu/read/2312/chapter/6. [Accedido: 10-jun-2019].

[2] H.-B. L. Art Astrin, D. L. Ranjeet Patro, A. B. Jin-Meng Ho, S.-H. (Shannon) P. Marco Hernandez, y Igor Dotlic, «IEEE Standard for Local and metropolitan area networks - Part 15.6: Wireless Body Area Networks», *IEEE Std 802.15.6-2012*, pp. 1-271, feb. 2012.

[3] M. Toorani, «Security analysis of the IEEE 802.15.6 standard», *Int J Commun Syst*, vol. 29, n.º 17, pp. 2471-2489, nov. 2016.

[4] J. Jonsson, «On the Security of CTR + CBC-MAC», en *Selected Areas in Cryptography*, 2003, pp. 76-93.

[5] «CBC-MAC Definition from PC Magazine Encyclopedia», *PC*. [En línea]. Disponible en: https://www.pcmag.com/encyclopedia/term/39348/cbc-mac. [Accedido: 08-nov-2019].

[6] D. Whiting, N. Ferguson, y R. Housley, «Counter with CBC-MAC (CCM)». [En línea]. Disponible en: https://tools.ietf.org/html/rfc3610. [Accedido: 06-jun-2019].

[7] G. A. Dávila Vintimilla y O. D. Torres Sánchez, «Evaluación de vulnerabilidades del protocolo Display Authenticated Association, definido en el estándar IEEE 802.15.6, mediante la implementación de un escenario de pruebas», EPN, Quito, 2018.

[8] M. J. Dworkin, «Recommendation for block cipher modes of operation :: the CMAC mode for authentication», National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-38b, 2016.

[9] «NIST, Advanced Encryption Standard (AES), Sprinfield: Federal Information Processing Standards Publications». 2001.

[10] M. Li, W. Lou, y K. Ren, «DATA SECURITY AND PRIVACY IN WIRELESS BODY AREA NETWORKS», *IEEE Wireless Communications*, p. 8, 2010.

[11] X. Lai, Q. Liu, X. Wei, W. Wang, G. Zhou, y G. Han, «A Survey of Body Sensor Networks», *Sensors (Basel)*, vol. 13, n.º 5, pp. 5406-5447, abr. 2013.

[12] G.-Z. Yang, Ed., *Body sensor networks*, 2. ed. London: Springer, 2014.

[13] D. P. Tobón, T. H. Falk, y M. Maier, «Context awareness in WBANs: a survey on medical and non-medical applications», *IEEE Wireless Communications*, vol. 20, n.º 4, pp. 30-37, ago. 2013.

[14] Md. T. Arefin, M. H. Ali, y A. K. M. F. Haque, «Wireless Body Area Network: An Overview and Various Applications», *JCC*, vol. 05, n.º 07, pp. 53-64, 2017.

[15] S. Pathania y N. Bilandi, «SECURITY ISSUES IN WIRELESS BODY AREA NETWORK», 2014.

[16] M. Al Ameen, J. Liu, y K. Kwak, «Security and Privacy Issues in Wireless Sensor Networks for Healthcare Applications», *J Med Syst*, vol. 36, n.º 1, pp. 93-101, feb. 2012.

[17] «Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de confidencialidad», en *Redes de Datos y Comunicaciones entre Sistemas Abiertos*, Recomendacion UIT-T X.814, 1996.

[18] «Tecnología de la Información – Interconexión de Sistemas Abiertos – Marcos de Seguridad para Sistemas Abiertos: Marco de Autenticación», en *Redes de Datos y Comunicaciones entre Sistemas Abiertos*, Recomendacion UIT-T X.811, 1996.

[19] «Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de Integridad», en *Redes de Datos y Comunicaciones entre Sistemas Abiertos*, Recomendacion UIT-T X.815, 1996.

[20] «Requisitos de seguridad para las redes de telecomunicaciones», en *Explotación General de la Red, Servicio Telefónico, Explotación del Servicio y Factores Humanos*, Recomendacion UIT-T E.408, 2004.

[21] «ARQUITECTURA DE SEGURIDAD DE LA INTERCONEXIÓN DE SISTEMAS ABIERTOS PARA APLICACIONES DEL CCITT», en *REDES DE COMUNICACIÓN DE DATOS: INTERCONEXIÓN DE SISTEMAS ABIERTOS (ISA); SEGURIDAD, ESTRUCTURA Y APLICACIONES*, Recomendacion UIT-T X.800, 1991.

[22] «Information technology. Security techniques. Entity authentication. Part 1: General», en *ISO/IEC 9798-1*, 2010.

[23] «Código de autenticación de mensajes: descripción general | Temas de ScienceDirect», *ScienceDirect*. [En línea]. Disponible en: https://www.sciencedirect.com/topics/computer-science/message-authentication-code. [Accedido: 09-nov-2019].

[24] J. E. Averos Vargas, «Estudio del Estándar IEEE 802.15.6 y Simulación de los Parámetros de Transmisión en una Red de Área Corporal en la Banda de Frecuencia de 2.4 GHz.», EPN, Quito, 2017.

[25] J. F. Villegas Méndez, «Comunicaciones WBAN-IBC: enfoques, perspectivas y aplicacciones.», Universidad Politécnica de Valencia, 2017.

[26] S. Ullah, M. Mohaisen, y M. A. Alnuem, «A Review of IEEE 802.15.6 MAC, PHY, and Security Specifications», *International Journal of Distributed Sensor Networks*, vol. 9, n.º 4, p. 950704, abr. 2013.

[27] M. Hernandez y R. Miura, *Body Area Networks using IEEE 802.15.6: Implementing the ultra wide band physical layer*. Academic Press, 2014.

[28] M. J. Dworkin *et al.*, «Advanced Encryption Standard (AES)», *Federal Inf. Process. Stds. (NIST FIPS) - 197*, 26-nov-2001. [En línea]. Disponible en: https://www.nist.gov/publications/advanced-encryption-standard-aes. [Accedido: 19-jul-2019].

[29] L. A. Pousa, «ALGORITMO DE CIFRADO SIMÉTRICO AES. ACELERACIÓN DE TIEMPO DE CÓMPUTO SOBRE ARQUITECTURAS MULTICORE.», p. 41.

[30] «Recommendation for Block Cipher Modes of Operation. Methods and Techniques», en *NIST Special Publication 800-38A*, 2001.ª ed., .

[31] JH. Song, R. Poovendran, J. Lee, y T. Iwata, «The AES-CMAC Algorithm», RFC Editor, RFC4493, jun. 2006.

[32] «Qué es el lenguaje unificado de modelado (UML)», *Lucidchart*. [En línea]. Disponible en: https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml. [Accedido: 05-sep-2019].

ANEXOS

ANEXO A. TABLAS DE TRAMAS.

ANEXO B. DIAGRAMAS DE SECUENCIA.

ANEXO C. DIAGRAMAS DE CLASE.

ANEXO D. CÓDIGO DEL PROGRAMACIÓN (INCLUIDO EN CD).

ANEXO E. ESTÁNDAR IEEE 802.15.6 (INCLUIDO EN CD).

ORDEN DE EMPASTADO