UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

Tema:

"HACKING ÉTICO AL IOT MEDIANTE SDR"

Trabajo de Graduación. Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Electrónica y Comunicaciones.

SUBLÍNEA DE INVESTIGACIÓN: Seguridad de la Información

AUTOR: Carlos Andrés Valencia Llerena

TUTOR: Ing. Santiago Manzano, Mg.

Ambato - Ecuador

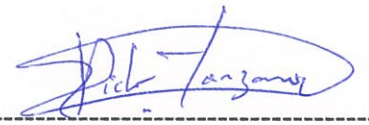
Octubre 2018

APROBACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Investigación sobre el tema: "HACKING ÉTICO AL IOT MEDIANTE SDR.", del señor, CARLOS ANDRÉS VALENCIA LLERENA, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el numeral 7.2 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ambato, 17 de Octubre del 2018

EL TUTOR



Ing. Santiago Manzano, Mg.

AUTORÍA

El presente Proyecto de Investigación titulado: "HACKING ÉTICO AL IOT MEDIANTE SDR.", es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, 17 de Octubre del 2018

Carlos Andrés Valencia Llerena
CC: 1804453130

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad.

Ambato, 17 de Octubre del 2018

Carlos Andrés Valencia Llerena
CC: 1804453130

APROBACIÓN DE LA COMISIÓN CALIFICADORA

La Comisión Calificadora del presente trabajo conformada por los señores docentes PhD. Carlos Gordon e Ing. Mg. Marco Jurado, revisó y aprobó el Informe Final del Proyecto de Investigación titulado "HACKING ÉTICO AL IOT MEDIANTE SDR.", presentado por el señor Valencia Llerena Carlos Andrés de acuerdo al numeral 9.1 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ing. Mg. Elsa Pilar Urrutia Urrutia
PRESIDENTA DEL TRIBUNAL

PhD. Carlos Gordón
DOCENTE CALIFICADOR

Ing. Mg. Marco Jurado
DOCENTE CALIFICADOR

DEDICATORIA:

A mi madre Olga Valencia por ser el pilar fundamental en mi vida, hizo todo lo posible por darme lo mejor, admiro su esfuerzo y sacrificio día a día para con sus tres hijos, me inculco buenos valores, y será por siempre mi modelo seguir. A mis hermanas Gaby y Salomé por compartir a mi lado y apoyarme en todo.

Carlos Valencia

AGRADECIMIENTO:

A Dios que es mi guía por responder a cada una de mis oraciones me ayudo a tomar las mejores decisiones, todo se lo debo a Él.
A mi madre que desde pequeño me dio lo mejor cumplió doble rol en mi vida me siento orgulloso de tenerla junto a mí.
A mi tutor Ing Santiago Manzano por su ayuda en este trabajo y a mis amigos con los que compartí buenos momentos.

Carlos Valencia

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN EJECUTIVO

El presente proyecto de investigación desarrolla un modelo de hacking ético a las diferentes tecnologías inalámbricas orientadas al Internet de las cosas (IoT) como son Bluetooth, Wifi, Z wave. Este estudio se construye en base a 4 etapas: acceso, transferencia, intrusión y replica de señal, en la primera fase se capta la señal de las aplicaciones desarrolladas donde el uso de radio definido por software (SDR) desempeña un papel fundamental como herramienta principal que ejemplifica los dispositivos reales de sistemas de comunicaciones inalámbricos orientados al IoT mediante diagramas de bloques diseñados en la herramienta GNU radio companion, en donde se ingresan y modifican características propias de la señal como frecuencia, ganancia, modulación, etc. Además de tratar a la señal mediante amplificadores, filtros y bloques de soporte de señal que protegen al equipo principal de sobrecargo de energía., en la segunda fase se envía la información obtenida que contiene: direcciones Mac de los dispositivos maestro – esclavo, canales de acceso, modos de comunicación y payload a la herramienta Wireshark para posteriormente analizarla, en la fase tres se muestran los datos en forma de trama que representan los principales protocolos que componen las tecnologías antes mencionadas, finalmente se ejecuta la réplica de señal en Z wave por ser la única que no está encriptada. El conjunto de etapas ya descritas que desarrollan el proceso de hacking ético tienen como base de ejecución sistemas de código abierto tanto en hardware como en software.

Palabras claves.- Hacking ético, SDR, Gnu radio Companion, IoT, Trama.

ABSTRACT

This research project develops a model of ethical hacking to the different wireless technologies oriented to the Internet of Things (IoT), such as Bluetooth, Wifi, Z wave. This study builds on 4 stages: access, transfer, intrusion and replica of signal, in the first phase captures the signal of the applications developed where the use of software defined radio (SDR) plays a key role as the main tool that exemplifies the actual devices of wireless communications systems oriented to the IoT using block diagrams designed in GNU radio companion, where you enter and modify characteristics of the signal as a frequency, gain, modulation, etc. In addition to treating the signal using amplifiers, filters and blocks of sign bracket that protect the core team of surcharge of energy., in the second phase sends the information obtained that contains: MAC addresses of the devices Master - Slave, access channels, modes of communication and payload to the tool Wireshark and later analyze it, in phase three, the data is displayed in the form of a frame that represent the main protocols that make up these technologies, finally runs the replica signal in Z wave for being the only one that is not encrypted. This set of stages that develop the process of ethical hacking have as the basis for implementing open source systems in both hardware and software.

Key words.- Ethical Hacking, SDR, Gnu radio Companion, IoT, Frame.

GLOSARIO DE TÉRMINOS Y ACRÓNIMOS

SDR.- Radio definida por software o en inglés “Software Defined Radio (SDR) es un sistema de comunicación por radio donde los componentes que han sido normalmente implementada en el hardware como mezcladores, filtros, amplificadores, moduladores/demoduladores, detectores, que se implementó en su lugar por medio de software en un ordenador personal o al sistema integrado.

GNU.- Gnu radio es un sistema operativo que consiste en varios programas fundamentales que necesita el ordenador para poder comunicar y recibir instrucciones de los usuarios; tales como leer y escribir datos en el disco duro, cintas, e impresoras; controlar el uso de la memoria; y ejecutar otros programas.

Hacking Ético.- Hacking ético es una disciplina profesional dentro del campo de la seguridad informática que permite evaluar el nivel de vulnerabilidad y el riesgo en el que se encuentran los sistemas informáticos o los activos de una organización mediante un acuerdo previo con el cliente.

Flujograma.- Flujograma es una representación gráfica de un proceso, ofrece una descripción visual de las actividades implicadas en un proceso.

Frecuencia Intermedia.- Frecuencia intermedia es la frecuencia que en los aparatos de radio que emplean el principio superheterodino se obtiene de la mezcla de la señal sintonizada en antena con una frecuencia variable generada localmente en el propio aparato mediante un oscilador local (OL) y que guarda con ella una diferencia constante.

IoT.- El internet de las cosas (IoT) es un concepto que se refiere a una interconexión digital de objetos cotidianos con internet.

Protocolo de comunicación.- Protocolo de comunicación es un sistema de reglas que permiten que dos o más entidades de un sistema de comunicación se comuniquen entre ellas para transmitir información por medio de cualquier tipo de variación de una magnitud física

INTRODUCCIÓN

En la era tecnológica actual el internet de las cosas se ha convertido en una realidad a medida que más y más dispositivos se conectan a Internet, la influencia del IoT en las actividades diarias del ser humano ha generado un gran impacto en desarrolladores de aplicaciones por lo que su diseño y construcción se ha incrementado llegando a una cifra aproximada de 25 mil dispositivos inteligentes según Howard Jensen investigador del Instituto médico, científico e industrial (ISM) ubicado Connecticut (USA) en lo que va de los últimos 5 años. Además, menciona que los fabricantes a menudo pasan por alto la seguridad informática enfocándose en marketing y competencia empresarial.

La aparición de sistemas de radio programable conocidos como SDR; reemplazando a modelos de comunicaciones basados en hardware están experimentando un rápido crecimiento tecnológico. Ofrecen flexibilidad, escalabilidad, seguridad y su utilidad en investigaciones ha permitido que se pueda evaluar y atacar diferentes tipos de sistemas inalámbricos a nivel de protocolo sin necesidad de invertir en costoso hardware o equipo para la fabricación de circuitos, irrumpiendo su seguridad considerablemente más fácil.

El Capítulo I describe las problemáticas que se hallan en la seguridad informática de dispositivos IoT, dichos elementos son de diferentes tecnologías (bluetooth, wifi, z wave) pero trabajan en una misma banda de frecuencia por lo que pueden ocasionar colapsos de red y de funcionamiento, además da una idea del número de dispositivos operando en la actualidad y sus posibles eventualidades.

En el Capítulo II se hace mención a trabajos investigativos relacionados a determinar vulnerabilidades de seguridad en sistemas IoT orientados a la industria, salud, economía, etc. Además, incluye la fundamentación teórica del proyecto de investigación

A continuación del marco teórico el Capítulo III detalla la metodología utilizada en el desarrollo de este proyecto de investigación.

En el Capítulo IV se describe el desarrollo de un proceso de hacking ético mediante radio definido por software a diferentes aplicaciones en tres tecnologías comunes en

el IoT para verificar vulnerabilidades de seguridad en dichos sistemas inalámbricos y cada uno de los pasos que llevaron al cumplimiento de los objetivos planteados.

Finalmente, en el Capítulo V se redactan las conclusiones y recomendaciones obtenidas del presente proyecto de investigación.

CAPÍTULO I

EL PROBLEMA

1.1. Tema de Investigación

"Hacking Ético al IoT mediante SDR"

1.2. Planteamiento del Problema

En la actualidad existen cientos de millones de usuarios de internet en el mundo y la cifra aumenta cada día según el ISM; incremento que va de la mano de un adelanto tecnológico que evoluciona a pasos agigantados. Este progreso en tecnología permite contar en la actualidad con 54 millones de dispositivos conectados a internet y en uso, número que seguirá aumentando y se estima que para el año 2020 existirán cerca de 61 millones de dispositivos IoT conectados; cifras que menciona Howard Jensen en un trabajo de investigación, por lo que será interesante ver si estos elementos inteligentes están en la capacidad de trabajar juntos en un espectro electromagnético limitado en comparación con las cifras antes mencionadas . A este conjunto de dispositivos conectados a internet operando unos con otros se le conoce como Internet de las Cosas [1].

Las redes inalámbricas orientadas al IoT trabajan en una misma frecuencia (2.4 GHz), por lo que los dispositivos electrónicos estarán operando bajo un entorno de gran cantidad de interferencias provocando solapamiento de señales entre los canales de comunicación reduciendo así su ancho de banda y su velocidad de trasmisión ocasionando un retraso en el envío de datos, además de reducir el desempeño del dispositivo provocando una colisión entre ellos [2]. Otro obstáculo para esta tecnología viene de aquellos que desarrollan aplicaciones IoT, generan herramientas simuladas, aplicables y totalmente funcionales trabajando de una manera autónoma, pero una vez que empiezan a operar en conjunto no cumplen las expectativas esperadas [3].

La seguridad en el IoT es el aspecto más importante a tomar en cuenta, una investigación realizada en 2016 por un investigador de la ISM revelo que gran parte de los dispositivos muestran problemas de privacidad en cuanto a seguridad informática, quedando vulnerables por: radio frecuencia o a través de la red IP, y fácilmente se podría clonar dispositivos manejarlos remotamente dejando en evidencia y a disposición ajena información personal de gran importancia generando serios inconvenientes no solo de manera personal sino también a un campo de alto interés como la industria generando pérdidas económicas si se filtra la información, en el área de la salud alterando resultados provocando fallos en casos clínicos de alto interés, desventajas en el trasporte privado; alteración en los requerimientos de usuario y cambios de ruta para fines antisociales, por esta razón los usuarios recurren a herramientas del pasado que garanticen mayor confiabilidad dejando a un lado la innovación y tecnología [4].

1.3. Delimitación

DELIMITACÍON DE CONTENIDOS

Área Académica de la carrera: Comunicaciones

Línea de Investigación: Tecnologías de Comunicación

Sublínea de Investigación: Seguridad de la Información

DELIMITACION ESPCIAL

El presente proyecto de investigación se desarrolló en la Universidad Técnica de Ambato, Facultad de Ingeniería en Sistemas, Electrónica e Industrial.

DELIMITACIÓN TEMPORAL

La presente investigación se desarrollará en el período Abril 2017 - Agosto 2018 de acuerdo a lo establecido en el Reglamento de graduación para obtener el título terminal de tercer nivel de la Universidad Técnica de Ambato.

1.4. Justificación

La investigación se enfoca en un análisis de información de señales de radio frecuencia en protocolos relacionados al IoT obtenida en un proceso de hacking, por medio de simulaciones de sistemas de comunicaciones basados en radio software. Este proceso permitirá profundizar los conocimientos en seguridad a nivel de protocolo en estas tecnologías IoT y brindar tratamiento de señales de manera práctica y en tiempo real.

El estudio de seguridad informática debería ser una prioridad, ya que hay miles de maneras en las que un individuo con malas intenciones puede hacerse con la información generando pérdidas económicas en el área industrial y filtrar datos de carácter personal a terceros, tomando en cuenta la importancia del tipo de datos que manejan estos dispositivos conectados a la red de internet la prioridad de los desarrolladores debería concentrarse no solo en satisfacer en desempeño a los usuarios sino también proteger la integridad y la privacidad de la información situación que no se maneja con niveles críticos de interés , poniendo énfasis en el marketing y la competencia empresarial.

Al sustentar la investigación y poner en evidencia los fallos en la seguridad a nivel de protocolo de red los beneficiarios son todos aquellos vinculados a la tecnología IoT: usuarios que contaran con alternativas orientadas a aplicaciones con arquitectura simple pero con niveles de seguridad confiables, la industria garantizando seguridad en la información por contar con aplicaciones robustas a nivel físico lógico desarrolladores de aplicaciones IoT con pautas de seguridad para el diseño de nuevos dispositivos y productores de dichos dispositivos que garanticen confiabilidad en los productos a ofrecer al mercado, a más de ser una pauta para futuros proyectos e investigaciones.

1.5. Objetivos

1.5.1 Objetivo General

- ✓ Aplicar un Hacking ético al IoT mediante SDR

1.5.2 Objetivos Específicos

- ✓ Analizar las tecnologías del Internet de las Cosas

- ✓ Diseñar aplicaciones para el análisis de protocolos de radio frecuencia utilizados en el internet de las cosas

- ✓ Desarrollar un prototipo de hacking ético para un sistema IoT

CAPÍTULO II

MARCO TEÓRICO

2.1 Antecedentes Investigativos

A continuación, se describen algunas publicaciones relacionadas al tema de investigación:

En el año 2015 Mark Stanislavc y Tod Beardsley publican un artículo en la revista Rapid7 "HACKING IOT: A CASE STUDY ON BABY MONITOR EXPOSURES AND VULNERABILITIES" donde describen las vulnerabilidades y exposiciones comunes de los dispositivos IoT mediante Air Magnet Survey herramienta simulada basada en diagramas de calor. Los dispositivos IoT sufren problemas de software, firmware y hardware, pero es raro encontrar un dispositivo IoT que no exhiba al menos un fallo crítico. Por tanto exhibieron varias vulnerabilidades y exposiciones comunes de los monitores para bebes como: interferencias, fácil encriptación a la comunicación, acceso mediante comunicaciones remotas, etc. Dejando en evidencia un nivel de seguridad bajo en estos equipos [2].

Jack L. Ziegler; Robert T. Arn; William Chambers en un artículo publicado en el año 2017 en la IEEE realizaron una investigación "MODULATION RECOGNITION WITH GNU RADIO, KERAS, AND HACKRF" donde se toma como referencia el conocimiento previo de los datos recibidos de una señal, como potencia, frecuencia y fase, degradación de señal e interferencia, la tarea es reconstruir la información enviada sin errores utilizando dos transceptores simples y el uso de sistemas operativos abiertos (Keras) que hagan el papel de demodulador, con esto se pudo crear un conjunto de redes neuronales para diferentes tipos de datos y la recuperación de la señal no fue tarea compleja [4].

Deepak Vohra; Arusha Dubey; Khyati Vachhhani en un artículo publicado en la IEEE en el año 2016 “INVESTIGATING GSM CONTROL CHANNELS WITH RTL-SDR AND GNU RADIO” Centran el cumplimiento de las especificaciones GSM con el uso de gráficos de flujo programables proporcionados por GNU, con esto se observó que las torres celulares utilizaban tanto el salto de frecuencia como el cifrado, al momento de enlazar una llamada mediante el conteo de canal y el ancho de banda en uso mediante el dispositivo móvil de red respectiva. La información de canal así obtenida proporciona técnicas eficientes para la detección de agujeros de espectro que proporcionan una alta capacidad de resolución espectral [5].

Jesús Molina y Joe Gordon en una conferencia otorgada en el año 2016 presentaron un artículo llamado “HACKING THE IOT: WHEN GOOD DEVICES GO BAD” en el cual hablan específicamente de la nueva tecnología además de proponer un sistema conformado por software libre que les permite realizar pruebas de vulnerabilidades de seguridad a los dispositivos de una red de mediante GNU radio. Sus resultados fueron nada más que la confirmación de bajos niveles de seguridad en los sistemas IoT [3].

2.2 Fundamentación Teórica

2.2.1 Redes inalámbricas

Las redes inalámbricas se comunican por medios no guiados a través de ondas electromagnéticas donde la transmisión y recepción se realiza por medio de antenas. El uso de este tipo de redes es la única solución para zonas a las que no llega el cableado, como es el caso de zonas rurales, aunque presenta desventajas como la susceptibilidad a cambios atmosféricos y la falta de seguridad en el intercambio de información [6].

Existen varios tipos de redes inalámbricas los cuales se distinguen según su alcance y tecnología, estos tipos de redes se diferencian por sus áreas de cobertura, las tecnologías y estándares que manejan, tal como se muestra en la Figura 1:

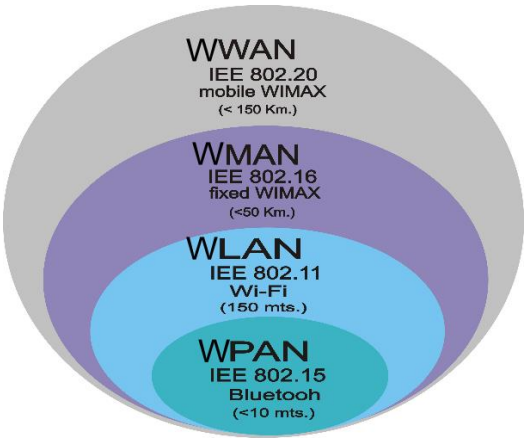


Figura 1 Tipos de redes inalámbricas [7]

2.2.2 El internet de las cosas

El internet de las cosas es una red de objetos conectados a Internet capaces de recolectar e intercambiar datos usando sensores integrados, el IoT hace referencia a cualquier dispositivo autónomo conectado que se pueda monitorear o controlar desde una ubicación remota y gestionar las cosas cotidianas con la tecnología para mejorar la calidad de vida del ser humano, la figura 2 muestra un esquema IoT que contiene una red de banda ancha (voz, video y datos) donde la información es gestionada por la nube [8].

Figura 2 Esquema de un entorno IoT [8]

El IoT incluye diferentes sistemas como:

- Coches conectados a Internet
- Wearables, incluyendo dispositivos de salud y fitness, relojes, y dispositivos implantados en humanos.
- Medidores y objetos inertes en general, que se dotan de inteligencia
- Sistemas de automatización y control para inmuebles e iluminación de Smartphone
- Redes de sensores inalámbricos que captan información del tiempo, barreras anti-inundaciones, mareas y muchas otras aplicaciones.

2.2.3 Características del IoT.

La combinación de software y hardware en el IoT proporciona la facilidad de convertir un dispositivo habitual en inteligente, mostrando las siguientes características. [8]

- **Conectividad.-** La conectividad permite compatibilidad y acceso a la red, sea cual sea el medio que le rodea.
- **Sensibilidad.-** La sensibilidad permite detección y reconocimiento que reflejen un verdadero conocimiento del mundo físico y sus habitantes.
- **Interacción.-** La interacción permite comunicación entre el mundo físico, las personas y las máquinas, productos que interactúan de forma inteligente con el mundo real.

2.2.4 Arquitectura IoT

Se puede definir a la arquitectura de red como un sistema funcional compuesto de equipos de transmisión, programas y protocolos de comunicación que permite la transmisión de datos entre los diferentes componentes. Sin embargo no existe un modelo fijo que pueda definirse como propia al IoT, por lo tanto en base a su concepto la arquitectura que adopte tiene que cumplir ciertos requerimientos para que esta tecnología sea viable. Debe permitir que la tecnología sea distribuida, escalable, eficiente y segura donde los objetos puedan interactuar entre ellos [9].

La estandarización de una arquitectura para la IoT es aún un proceso en desarrollo, la tendencia ha estado inclinada a dar solución a dos problemas fundamentales: buscar

una forma estándar de acceso al medio y a los dispositivos, y buscar la forma de integrar los dispositivos a la Internet.

Tomando como base a las propuestas anteriores y haciendo una analogía a los conocidos modelos OSI y TCP/IP, esta tendencia a la estandarización puede quedar resumida como se muestra en la figura 3 [10]:

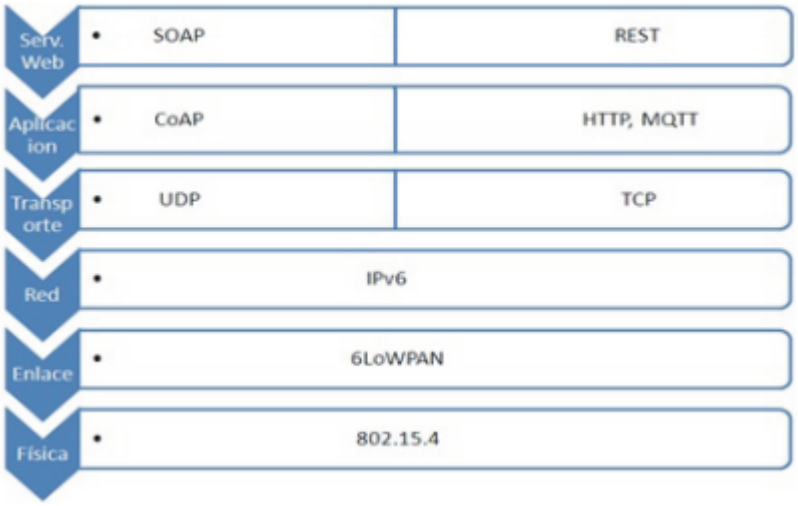


Figura 3 Stack de protocolos de la arquitectura del IoT [10]

2.2.5 Modelos de Comunicación para el IoT

Los modelos de comunicación para Internet de las Cosas son referencias que describen cómo se distribuyen e interactúan los diferentes protagonistas que forman una red de varios dispositivos conectados. De forma más o menos consensuada según el IETF, se han definido cuatro modelos de comunicación que pueden ser utilizados en el desarrollo de aplicaciones para Internet de las Cosas, los cuales son:

Modelo de comunicación Dispositivo – Dispositivo.- Este modelo representa dos o más dispositivos que se conectan entre ellos y la comunicación se establece directamente entre cada dos como se muestra en la figura 4. No hay ningún intermediario que gestione la comunicación ni enrute los datos ni nada por el estilo. Esta comunicación lógicamente sólo es posible si los dispositivos son capaces de entenderse en términos de interoperabilidad. El modelo de comunicación dispositivo a dispositivo encaja bien en aplicaciones de domótica o telemetría, en las que los dispositivos intercambian pequeños paquetes de datos a baja velocidad y cada cierto tiempo.

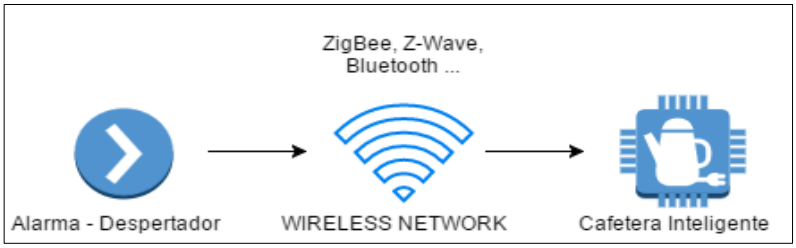


Figura 4 Modelo de comunicación Dispositivo – Dispositivo [11]

Modelo de comunicación Dispositivo – Internet.- En este modelo encajan todas aquellas aplicaciones en las que el dispositivo IoT tiene capacidad suficiente para conectar directamente con algún servicio en la nube, con el que comparte datos para analizarlos por algoritmos o enviar comandos de control para actuar directamente sobre el dispositivo como se muestra en la figura 5. Para que esto sea posible, el dispositivo debe de contener hardware específico para poderse conectar vía Ethernet o Wifi y tener recursos suficientes como para albergar una pila TCP/IP.

Intervienen dispositivos cerrados y poco o nada se puede hacer para programar una propia aplicación. Incluso en muchas ocasiones, los fabricantes de dispositivo son los que proporcionan un servicio propietario en la nube, sujeto a cuatro funcionalidades extra para ampliar el desempeño del dispositivo.

Figura 5 Modelo de comunicación Dispositivo – Internet [11]

Modelo de comunicación Dispositivo – Gateway.- Este es el modelo de comunicación más utilizado en Internet de las Cosas y se da cuando los dispositivos se conectan a Internet a través de un elemento que hace las veces de Gateway. Para ser más preciso, el gateway está dotado con un software de aplicación que hace de intermediario entre los dispositivos IoT y los servicios en la nube, haciendo principalmente de traductor entre protocolos como se muestra en la figura 6.

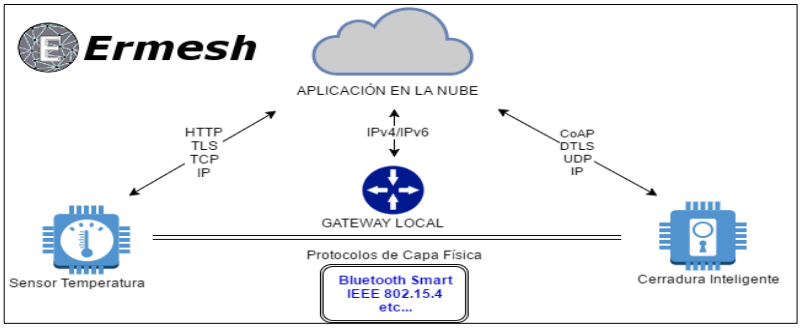


Figura 6 Modelo de comunicación Dispositivo – Internet [11]

Hay varios dispositivos que se utilizan hoy en día como gateway. Uno de los más populares y flexibles son los smartphones, capaces de conectar dispositivos IoT e Internet a través de alguna app y aprovechando la conexión a Internet que traen de serie. Por ejemplo, esta estrategia se está poniendo muy de moda en el mundo fitness para conectar sensores de frecuencia cardíaca, running y otros, con aplicaciones de seguimiento de la actividad en la nube.

Otro dispositivo en auge con capacidades de gateway son los hubs propietarios que están saliendo para aplicaciones de domótica. Suelen ser dispositivos capaces de conectarse con los dispositivos IoT mediante diferentes tecnologías, normalmente inalámbricas, y en el lado Internet, los fabricantes de estos hubs proporcionan al usuario algún servicio propietario para el almacenamiento y procesamiento de los datos recogidos. Para que te hagas una idea, echa un vistazo a Smart Things. Es un gateway capaz de comunicar con dispositivos Z-Wave y Zigbee, y además ofrece al usuario acceder a los dispositivos mediante un servicio en la nube.

Modelo de comunicación Back End.- Este modelo permite a los usuarios recolectar datos de sus dispositivos IoT, analizarlos y tomar decisiones inteligentes. A menudo, los servicios en la nube permiten además portabilidad o compartir datos con otros servicios como se muestran en la figura 7.

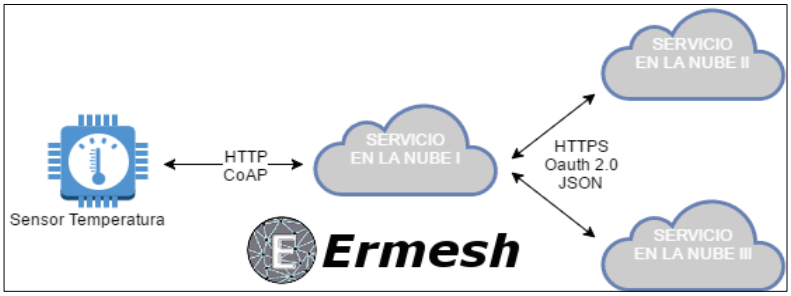


Figura 7 Modelo de comunicación Back End [11]

A nivel empresarial, está directamente relacionado con el auge del Big Data y el Business Intelligence. El objetivo principal de este modelo de arquitectura es conseguir interoperabilidad entre los diferentes servicios en la nube, en un mundo que tiende a compartir toneladas de gigabytes de datos de todo tipo por las redes, haciendo de este proceso invisible para el usuario [11].

2.2.6 Clases de Dispositivos principales del IoT

Existen tres tipos de dispositivos principales que permiten el desarrollo de un sistema IoT:

- Los dispositivos más pequeños son los controladores embedded de 8 bits System-On-Chip (SOC). Un buen ejemplo de este Open Source hardware es Arduino. Por ejemplo: Arduino Uno platform, este tipo no suelen llevar sistema operativo (SO).

- El siguiente nivel son los dispositivos con una arquitectura de 32 bits como los chips de Atheros y ARM. Muchas veces incluyen pequeños routers domésticos y derivados de estos. Normalmente estos dispositivos se basan en plataformas de Linux embedded, como OpenWRT u otros sistemas operativos embedded. En algunos casos, no corren ningún SO. Por ejemplo: Arduino Zero o Arduino Yun.

- Las plataformas IoT con más capacidad son los sistemas completos de 32 y 64 bits. Estos sistemas, como Raspberry Pi o BeagleBone, pueden correr varios SO como Linux o Android. En muchos casos, estos son Smartphone o algún tipo de dispositivo basado en tecnologías móviles. Estos dispositivos pueden

comportarse como Gateways o puentes para dispositivos más pequeños. Por ejemplo: un wearable que se conecta vía Bluetooth a un Smartphone o a una Raspberry Pi, es típicamente un puente para conectarse a Internet [12].

2.2.7 Reglas de la comunicación en dispositivos IoT: Modo físico

En el modo físico los dispositivos del IoT siguen un proceso por el cual la información fluye del medio físico a un medio virtual. Aunque existen algunos modelos para conformar un sistema IoT el principio es el mismo; este proceso se puede dividir en cuatro fases como se muestra en la figura 8:

Figura 8 Modo Físico de conexión IoT [12]

.

Dispositivos finales.- Dispositivo final es la parte visible a los usuarios. Dentro de este bloque se encuentra a los sensores, actuadores y el hardware necesario (dispositivo principal) para comunicar el mundo físico con el mundo virtual. Actualmente existe infinidad de componentes, sobre todo para el prototipado, que permiten comenzar a crear dispositivos del IoT.

Puntos de acceso.- Los puntos de acceso permiten la conectividad de las cosas, objetos o dispositivos a Internet. El objetivo fundamental es establecer una conexión entre los periféricos (dispositivos u objetos conectados) y la nube, pero también debe permitir conectividad entre ellos. Esta conexión tiene que ser segura, robusta, tolerante a fallos a fin de que recoja la información obtenida de los dispositivos y a la vez, se puedan gestionar.

A través de los gateway o interfaces de comunicación, los dispositivos estarán conectados entre ellos y con la nube. Sin embargo el inconveniente está en los diferentes ecosistemas que existen, demasiadas pasarelas, protocolos, aplicaciones, etc. Para resolver esta cuestión, debe existir un núcleo general que pueda comunicar con todos ellos.

Procesamiento de Datos.- El procesamiento de datos es eje central del IoT. El buen funcionamiento de un sistema con estas características dependerá de las capacidades en la gestión de estos datos y el uso inteligente que se haga de ellos. Por este motivo, un sistema del IoT debe ser capaz de recolectar información de los sensores, almacenarlos y analizarlos. En este punto las plataformas en la nube enfocadas a este sector tienen mucho que ver.

Aplicaciones.- Las aplicaciones sirven para poder manejar y visualizar la información, da lo mismo si son nativas o web. Gracias al uso de APIs y servicios web, cualquier tipo de aplicación se podrá conectar a los datos y mostrarlos a los usuarios. Además de visualizar los datos las aplicaciones tendrá la capacidad de modificar los parámetros para que los sistemas se comporten de una manera determinada [9].

2.2.8 Fases de la comunicación en dispositivos IoT: Modo lógico

El modo lógico permite que se establezca la comunicación, los dispositivos deben saber cómo comunicarse.

Figura 9 Modo lógico de conexión IoT [13]

La comunicación comienza con un mensaje, o información, que se debe enviar desde un origen hasta un destino regido por protocolos que deben respetarse para que el mensaje se envíe y comprenda correctamente como se muestra en la figura 9. Para que la comunicación sea satisfactoria, hay varios protocolos que deben actuar:

Codificación de los mensajes: La codificación es el proceso mediante el cual la información se convierte en otra forma aceptable para la transmisión. La

decodificación invierte este proceso para interpretar la información. El host emisor, primero convierte en bits los mensajes enviados a través de la red. Cada bit se codifica en un patrón de sonidos, ondas de luz o impulsos electrónicos, según el medio de red a través del cual se transmitan los bits. El host de destino recibe y decodifica las señales para interpretar el mensaje.

Formato y encapsulación del mensaje: Los formatos de los mensajes dependen del tipo de mensaje y el canal que se utilice para entregar el mensaje. Este mensaje se encapsula en una trama antes de mandarse a la red, proporcionando la dirección de destino y de origen.
El formato y el contenido de una trama están determinados por el tipo de mensaje que se envía y el canal que se utiliza para enviarlo. Los mensajes que no tienen el formato correcto no se pueden enviar al host de destino o no pueden ser procesados por éste.

Tamaño del mensaje: El tamaño del mensaje debe estar entre 64 bytes y 1518 bytes para ser comprendida por el receptor. Las restricciones de tamaño de las tramas requieren que el host de origen divida un mensaje largo en fragmentos individuales que cumplan estos requisitos. Esto se conoce como segmentación. Cada segmento se encapsula en una trama separada con la información de la dirección y se envía a través de la red. En el host receptor, los mensajes se des encapsulan y se vuelven a unir para su procesamiento e interpretación.

Temporización del mensaje:

Método de acceso: Es el método que los hosts de una red necesitan para saber cuándo comenzar a enviar mensajes y cómo responder cuando se produce algún error. Si dos dispositivos emiten a la vez se produce una colisión.

Control de flujo: El control de flujo es el proceso por el cual puede transmitir mensajes a una velocidad mayor que la que puede recibir y procesar el host de destino. Los hosts de origen y destino utilizan el control del flujo para negociar la temporización correcta, a fin de que la comunicación sea exitosa.
Tiempo de espera de respuesta: En el tiempo de respuesta los hosts de las redes también tienen reglas que especifican cuánto tiempo deben buscar una respuesta y qué deben hacer si se agota el tiempo de espera para la respuesta.

Opciones de entrega del mensaje:

Unicast: En el modelo Unicast el emisor envía un mensaje a un solo destinatario directamente.

Multicast: En el modelo Multicast el emisor envía un mensaje a varios destinatarios simultáneamente.

Broadcast: En el Broadcast el emisor envía un mensaje a todos los dispositivos de la red [13].

2.2.9 Seguridad en el IoT

La seguridad es uno de los aspectos más importantes en IoT. Los dispositivos están constantemente captando información personal y, por su naturaleza, llevando el mundo real a Internet y vice-versa. Esto permite contar con tres categorías de riesgos entorno a la seguridad informática en dispositivos IoT, estos son:

La seguridad de la nube.- En la seguridad en la nube las amenazas más urgentes vienen del entorno de la empresa o del entorno de la nube al que estos dispositivos inteligentes están conectados.

La seguridad del dispositivo.- En la seguridad del dispositivo miles de millones de dispositivos conectados va a aumentar el uso de las aplicaciones de software y de los datos que se encuentran en los recursos de las empresas y en los dispositivos de consumo, lo que implica nuevos puntos de ataque para los hackers maliciosos

Seguridad para no causar ningún daño físico en dispositivos electrónicos, como por ejemplo el mal uso de actuadores [12].

La gestión del ciclo de vida de la seguridad.- En cuanto a la gestión si bien a menudo se pasa por alto, es un elemento fundamental para una estrategia de seguridad digital robusta y de largo plazo. La seguridad no es una actividad de una sola vez, sino una parte en evolución del ecosistema del IoT. El agregado de nuevos dispositivos, el desmantelamiento del dispositivo al final de su vida útil, la integración del dispositivo en un nuevo ecosistema de la nube o viceversa, la gestión de la descarga de firmware/software seguros son actividades que necesitan una gestión completa de las identidades, las claves y los tokens [14].

En la actualidad muchos fabricantes centran sus esfuerzos en desarrollar las funcionalidades que tienen más acogida entre sus potenciales usuarios para lanzar el producto al mercado con la mayor celeridad posible, descuidando aspectos tan fundamentales como la cyber seguridad, por ejemplo:

- Incluyen usuarios y contraseñas de acceso por defecto y sin mecanismos que obliguen al usuario a cambiarlas por otras más seguras.
- Las páginas web de control y configuración son inseguras.
- No ofrecen cifrado en las comunicaciones.
- No hay configuraciones de seguridad.
- Falta de soporte y actualizaciones de los fallos de seguridad detectados tanto en el software de control como en el firmware de los dispositivos [15].

.

2.2.10 Protocolos de Comunicación en el IoT

Los protocolos de comunicación en el IoT requieren conectividad a la red y por ello se relaciona directamente con el protocolo IP, el uso de esta tecnología es fundamental para la IoT porque permite la interoperabilidad de los sistemas, en la figura 10 se indica una jerarquización de distintos protocolos de red:

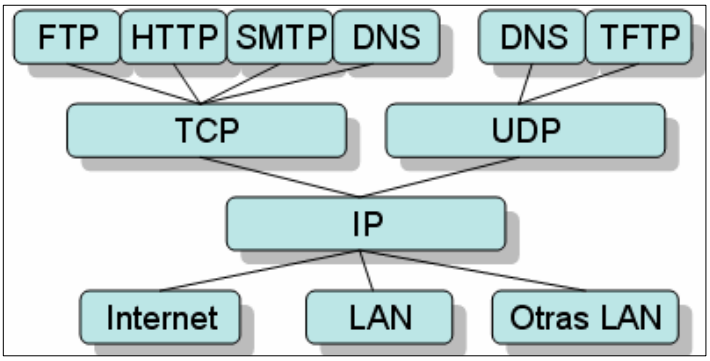


Figura 10 Protocolos de conexión relacionados al IoT [16]

Existen protocolos en cada capa o nivel de conexión, la capa inferior se refiere a la conectividad física que permite el desarrollo web, el bloque IP está destinado al enrutamiento y direccionamiento, la capa superior TCP – UDP, considerada como

protocolos de transporte, mientras que la capa final está vinculada con las aplicaciones que utiliza el usuario [16].

2.2.11 Protocolos IP usados en el IoT

Los protocolos IP permiten crear sistemas de comunicaciones IoT con las tecnologías web existentes: Hyper Text Transfer Protocol (HTTPS), Websockets o JavaScript Object Notation (JSON). A continuación se mencionan los protocolos más usados.

Protocolo HTTP.- Protocolo de Transferencia de Hipertexto. Es un sencillo protocolo cliente-servidor que articula los intercambios de información entre los clientes Web y los servidores HTTP. Se basa en sencillas operaciones de solicitud/respuesta. Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud. El servidor responde con un mensaje similar, que contiene el estado de la operación y su posible resultado. Todas las operaciones pueden adjuntar un objeto o recurso sobre el que actúan; cada objeto Web (documento HTML, fichero multimedia o aplicación CGI) es conocido por su URL.

Protocolo WebSocket.- El protocolo WebSocket proporciona comunicación dúplex completa a través de una conexión TCP única por la cual se pueden enviar mensajes entre el cliente y el servidor. Forma parte de la especificación HTML 5. El estándar WebSocket simplifica gran parte de la complejidad que circunda la comunicación web bidireccional y la administración de la conexión. Usar Websockets junto con HTTP es una solución apropiada para los dispositivos de IoT si los dispositivos pueden soportar las cargas de HTTP.

Protocolo XMPP.- Protocolo extensible de mensajería y presencia es un excelente ejemplo de una tecnología Web existente que encuentra un uso nuevo en el espacio de IoT. El XMPP tiene sus raíces en la mensajería instantánea y la información de presencia, y se ha ampliado a llamadas de voz y video, colaboración, middleware ligero, redifusión de contenido y enrutamiento generalizado de datos XML. Es un aspirante a administración a escala masiva de electrodomésticos de consumo, como lavadoras, secadoras, refrigeradores, etc. Las ventajas del XMPP son su direccionamiento, seguridad y escalabilidad. Esto lo hace ideal para aplicaciones de IoT orientada a los consumidores [17].

Otra forma de implementar sistemas de comuniones IoT es adaptar Internet a dispositivos electrónicos escalables en tecnología, por lo que empresas como Java, Genome, etc han lanzado algunos protocolos mencionados a continuación:

Protocolo CoAp.- El CoAP está semánticamente alineado con HTTP e, incluso, tiene asignaciones uno a uno hacia y desde HTTP. Los dispositivos de red están restringidos por microcontroladores más pequeños, con pequeñas cantidades de memoria flash y RAM, mientras que las restricciones en las redes locales, como 6LoWPAN, se deben a altas tasas de error de paquete y a un bajo rendimiento.

El CoAP puede ser un protocolo apropiado para dispositivos que operan con batería o mediante extracción de energía. Se muestran algunas características de este protocolo:

- Debido a que el CoAP usa UDP, algunas de las funciones de TCP se reproducen directamente en el CoAP. Por ejemplo, el CoAP distingue entre mensajes que se pueden confirmar (que requieren una confirmación) y que no se pueden confirmar.

- Las solicitudes y las respuestas se intercambian de forma asincrónica en los mensajes de CoAP (a diferencia del HTTP, donde se utiliza una conexión TCP existente).

- Todos los encabezados, métodos y códigos de estado se codifican de forma binaria, lo que reduce la sobrecarga del protocolo. Sin embargo, esto requiere el uso de un analizador de protocolo para solucionar problemas de red.

- A diferencia del HTTP, la capacidad de copiar en caché las respuestas de CoAP no depende del método de solicitud, sino del Código de respuesta.

Protocolo MQTT.- El protocolo de transporte de telemetría de cola de mensajes (MQTT) es de código abierto que se desarrolló y optimizó para dispositivos restringidos y redes de bajo ancho de banda, alta latencia o poco confiables. Es un transporte de mensajería de publicación/suscripción que es extremadamente ligero e ideal para conectar dispositivos pequeños a redes con ancho de banda mínimo.

El MQTT es eficiente en términos de ancho de banda, independiente de los datos y tiene reconocimiento de sesión continua, porque usa TCP. Tiene la finalidad de minimizar los requerimientos de recursos del dispositivo y, a la vez, tratar de asegurar la confiabilidad y cierto grado de seguridad de entrega con calidad del servicio.

El MQTT se orienta a grandes redes de dispositivos pequeños que necesitan la supervisión o el control de un servidor de back-end en Internet. No está diseñado para la transferencia de dispositivo a dispositivo. Tampoco está diseñado para realizar "multidifusión" de datos a muchos receptores. Este protocolo es simple y ofrece pocas opciones de control. Las aplicaciones que usan MQTT, por lo general, son lentas en el sentido de que la definición de "tiempo real" en este caso se mide habitualmente en segundos [17].

2.2.12 Gnu Radio

Gnu Radio es una plataforma de desarrollo de software gratuito que proporciona funciones de procesamiento de señales para la aplicación de las radios definidas por software. Brinda un acceso a un conjunto diverso de proyectos existentes centrado en la investigación de comunicaciones inalámbricas e implementación de sistemas de radio del mundo real [18].

Los diseños realizados en GNU Radio se programan en Python; un lenguaje de programación orientado a objetos, es decir, no se compila sino que el sistema operativo lo ejecuta directamente. Habitualmente esta plataforma de desarrollo (Gnu Radio) se compara con TCL, Perl, Scheme, Java y Ruby; sin embargo el uso de múltiples librerías en el procesamiento de señales permiten utilizar la aplicación GNU Radio como una herramienta que mediante una interfaz gráfica basada en bloques, similar a Simulink en Matlab, facilita la labor de diseño, creando los ficheros a partir del esquemático, de esta forma el usuario no tiene por qué saber Python.

Para construir un sistema de radio con GNU Radio se debe crear un grafo, donde los nodos son bloques de procesado de señal y los enlaces entre nodos representan el flujo de datos entre ellos. Los bloques de procesado de señal son implementados en C++ y portados a Python a partir del programa SWIG, es decir, para implementar los diseños se crearán los módulos en Python que a su vez se apoya en los bloques implementados

en C++. Las interacciones entre los diferentes niveles de GNU Radio se indican en la figura 11: [19]

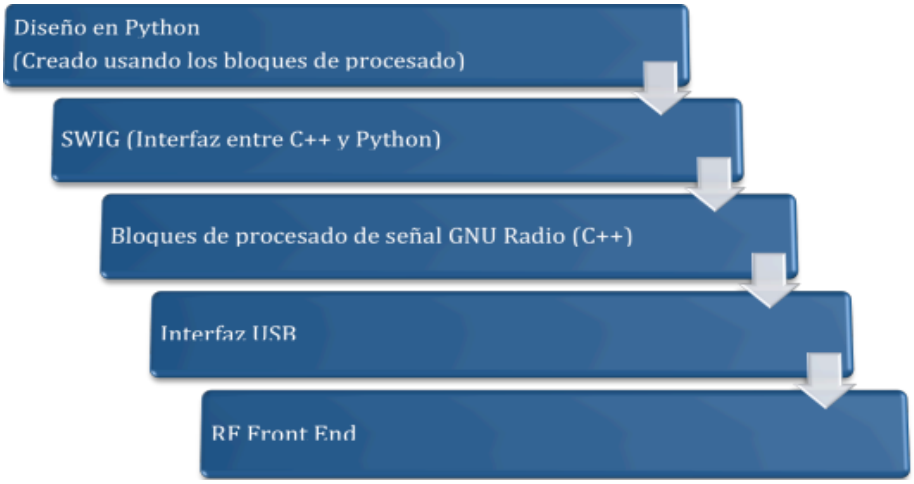


Figura 11 Esquema de radio con GNU Radio basado en Python [19]

Conceptualmente un bloque procesa señales, de forma continua, desde puertos de entrada hasta puertos de salida. Las partes de un bloque son el número de puerto de entrada, el número del puerto de salida, y el tipo de dato que fluirá de uno al otro.

2.2.13 Gnu Radio Companion

Gnu Radio Companion es una herramienta gráfica para realizar diseños a partir de bloques incluidos en librerías o creados por el usuario. Una vez realizado el diseño, esta plataforma de desarrollo genera el código en Python asociado, que bien puede ejecutarse desde la propia herramienta o desde la línea de comandos en un terminal. Para aquellos que sean usuarios de Matlab, cabe decir que la filosofía de funcionamiento es similar a la de Simulink. El GNU Radio Companion permite diseñar sistemas que utilicen un SDR como el USRP, pero también permite trabajar con otras señales, como por ejemplo las señales de audio del PC/Workstation.

La construcción de módulos utilizando esta herramienta gráfica se basa en añadir bloques e interconectarlos de manera esquemática. Los bloques utilizados se pueden agrupar de la siguiente manera:

Sources (fuentes): Sources estos bloques especifican cualquier tipo de fuente como por ejemplo un archivo de audio wav, un generador de señal de forma arbitraria con

las características que se requieran, una fuente binaria aleatoria, un fichero de cualquier formato, un micrófono, el propio universal software radio peripheral (USRP), etc.

Bloques de procesado de señal: Los bloques de procesado realizan un tratamiento de la señal de cualquier tipo. Se pueden citar como ejemplos los moduladores, filtros, remuestreadores, multiplicadores, amplificadores en software, etc.

Sinks (sumideros): En el sink la señal tendrá un destino final como bien puede ser un fichero de cualquier formato, la tarjeta de sonido o el USRP. A este conjunto también pertenecen los bloques de visualización de señales (Graphical sinks) como FFT sink (para visualizar la FFT de la señal en un punto), Constelation sink u Osciloscope sink (para representar la señal en tiempo en cualquier lugar del diseño) entre otros [19].

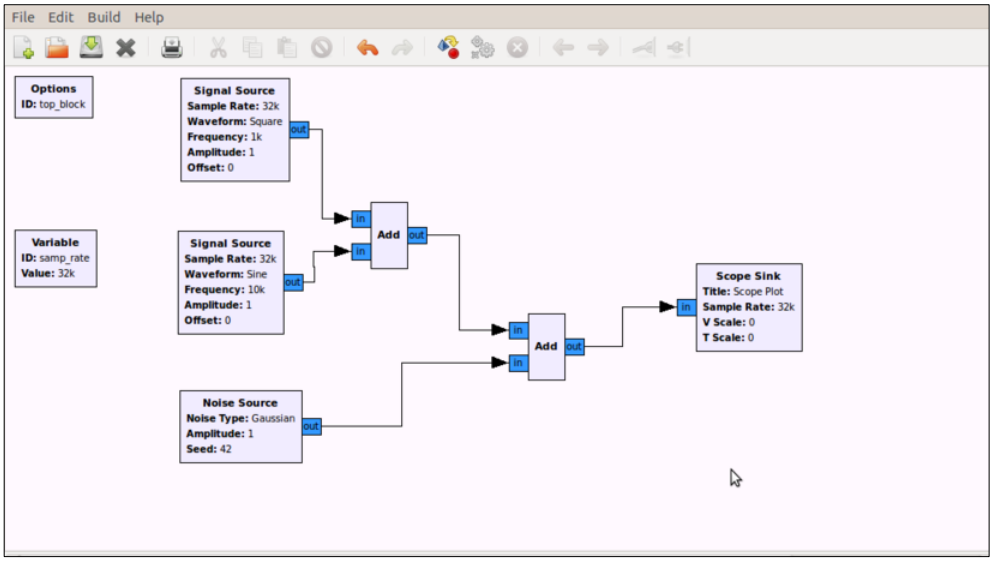


Figura 12 Adición de ruido a las señales de trasmisión utilizando GNU Radio [19]

En la figura 12 se indica un diagrama de bloques que muestra la adición de ruido a una señal (Fm), este esquema muestra las diferentes librerías mencionadas anteriormente con las que cuenta la plataforma de desarrollo Gnu Radio companion.

2.2.14 Radio definido por software

El radio definido por software es un sistema de radio comunicaciones en el cual una parte o la totalidad de sus componentes pertenecientes a la capa física (hardware) son implementadas por software. SDR mejora la interoperabilidad entre diferentes servicios, está compuesta de software y hardware, y puede ser reconfigurada para

habilitar comunicaciones entre una amplia variedad de normas de comunicaciones, protocolos y radio enlaces [12].

Análisis de las tecnologías IoT

2.2.15 Tecnología Bluetooth

La tecnología inalámbrica bluetooth se ha convertido en una especificación global de carácter tecnológico para el establecimiento de comunicaciones inalámbricas entre dispositivos portátiles, equipos de escritorio y periféricos. La tecnología bluetooth es de pequeña escala y de bajo costo, opera en la banda ISM de 2.4 GHz, universalmente disponible y que no requiere licencia [21].

Bluetooth se ha diseñado para operar en un ambiente multi usuario. Los dispositivos pueden habilitarse para comunicarse entre sí e intercambiar datos de una forma transparente al usuario. Hasta ocho usuarios o dispositivos pueden formar una piconet y hasta diez piconets pueden coexistir en la misma área de cobertura. Dado que cada enlace es codificado y protegido contra interferencia y pérdida de enlace, Bluetooth puede considerarse como una red inalámbrica de corto alcance muy segura [22].

Ventajas

- Tecnología ampliamente usada, especialmente en equipos y móviles de reciente producción
- Tecnología barata que puede reducir los costos de las empresas
- Está automatizado y no requiere configuración de conexión. Al estar con uno o más dispositivos en cobertura se comunican de forma automática.
- Los dispositivos bluetooth están normalizados, la compatibilidad entre diferentes modelos es segura y se podrán comunicar sin problemas
- Bajo consumo de energía ya que el uso de señales de baja potencia lo permite. Es recomendable para los dispositivos móviles
- Bluetooth puede actualizarse lo cual es ventajoso ya que vamos a poder ir actualizando las nuevas versiones del dispositivo. [23]

Desventajas

- Gasta mucha energía de la batería, cuando está en el modo visible sobre todo en telefonía móvil

- Cuando es usado inadecuadamente, podemos recibir mensajes y archivos indeseados

- Tiene un limitado radio de acción entre los periféricos ~100 metros entre ellos, la cantidad de dispositivos por piconet es otro inconveniente.

- No se recomienda el uso del Bluetooth para establecer conexiones a Internet debido a su limitada tasa de transferencia, una conexión LAN es más eficaz.

- La seguridad es otro factor desfavorable, sin embargo las versiones actuales contienen técnicas de encriptación mejoradas [24]

Arquitectura de Hardware

El hardware que compone el dispositivo Bluetooth está compuesto por dos partes: un dispositivo de radio, encargado de modular y transmitir la señal; y un controlador digital. El controlador digital está compuesto por un CPU, además de un procesador de señales digitales llamado controlador de enlace y de las interfaces con el dispositivo anfitrión como se muestra en la figura 13.

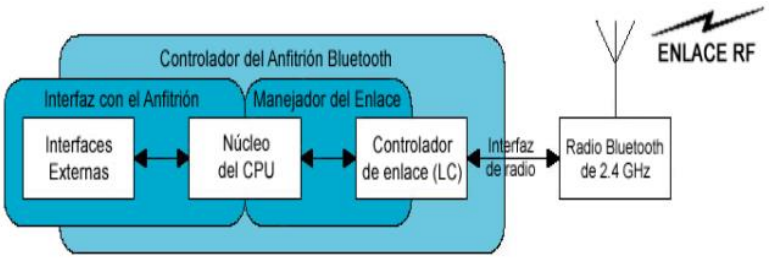


Figura 13 Arquitectura de Hardware [22]

El controlador de enlace está encargado de hacer el procesamiento de la banda base y del manejo de los protocolos ARQ y FEC de capa física. Además, se encarga de las funciones de transferencia (tanto asíncrona como síncrona), codificación de audio y inscripción de datos [22].

Arquitectura de Software

La arquitectura de software buscando ampliar la compatibilidad de los dispositivos Bluetooth, utiliza entre el dispositivo anfitrión y el dispositivo Bluetooth como tal una interfaz denominada HCI (Interface de control de host) como se muestra en la figura 14 [22].

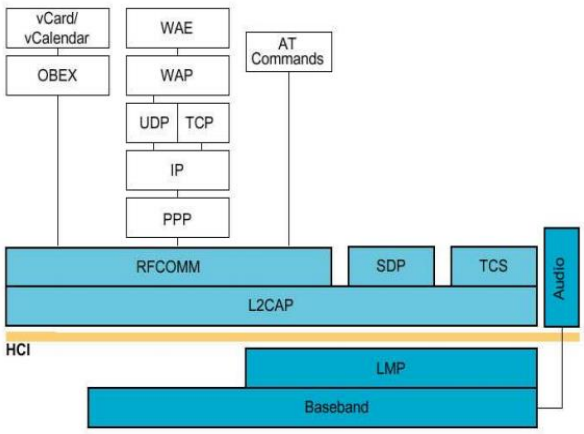


Figura 14 Arquitectura de Software [22]

La arquitectura de software está compuesta de diferentes protocolos de alto nivel como:

L2CAP.- Se encarga de la segmentación y reensamblaje de los paquetes para poder enviar paquetes de mayor tamaño a través de la conexión Bluetooth.

TCS.- Control de telefonía interactúa con el controlador de banda base a través del protocolo L2CAP (Controlador de enlace lógico y protocolo de adaptación).

SDP.- Utilizado para encontrar otros dispositivos Bluetooth dentro del rango de comunicación, encargado, también, de detectar la función de los dispositivos en rango.

Topología de Red Bluetooth

La topología de las redes Bluetooth puede ser punto-a-punto o punto-a-multipunto.

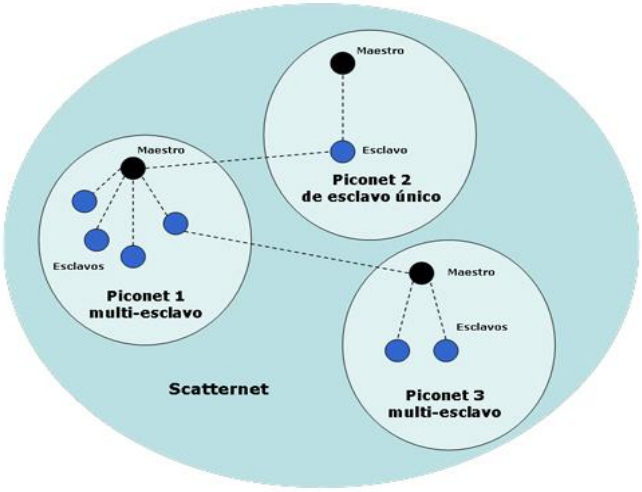


Figura 15 Topología de red Bluetooth [22]

Los dispositivos, se comunican en redes denominadas piconets. Estas redes tienen posibilidad de crecer hasta tener 8 conexiones punto a punto como se muestra en la figura 15. Además, se puede extender la red mediante la formación de scatternets [22].

Transmisión

La trasmisión bluetooth está diseñado para usar acuses de recibos y saltos de frecuencias, lo cual hará conexiones robustas. Esto está basado en paquetes, y saltarán a una nueva frecuencia después de que cada paquete es recibido, lo cual no solo ayuda a los problemas de interferencia, sino que añade seguridad. La tasa de datos es un megabytes/segundo, incluyendo el encabezado.

La transmisión de datos puede ser realizada de maneta síncrona o asíncrona. El método síncrono Orientado a Conexión (SCO) es usado principalmente para voz, y el asíncrono no orientado a conexión (ACL) es principalmente usado para transmitir datos [22].

Protocolo de conexión

Las conexiones Bluetooth, son establecidas a través de las siguientes técnicas. [22]

Standby.- Los dispositivos en una piconet que no están conectados, están en modo standby.

Page/Inquiry.- Al establecer una conexión con otro dispositivo, éste le envía un mensaje de tipo page, si la dirección es conocida; o una petición a través de un mensaje de page, si éste no es conocido. Estes método requiere una respuesta extra por parte del esclavo, desde la dirección MAC, que no es conocida por la unidad master.

Active.- Ocurre la transmisión de datos.

Hold.- Cuando el master o el slave desean, puede ser establecido un modo en el cual no son transmitidos datos. El objetivo de esto es conservar el poder..

Sniff.- Es aplicable solo para las unidades slaves, es para conservar el poder. Durante este modo, el slave no toma un rol ativo en la piconet, pero escucha a un reducido nivel.

Park.- Es un nivel más reducido, que el modo hold, el slave es sincronizado a la piconet pero no es parte del tráfico. En este estado, ellos no tienen direcciones MAC y solo escuchan para mantener su sincronización con el master y chequear los mensajes de broadcast

Corrección de Errores

Bluetooth tiene definidas tres técnicas de corrección de errores $\frac{1}{3}$ FEC, $\frac{2}{3}$ FEC y petición de repetición automática (ARQ). El propósito de los dos primeros es reducir el número de retrasmisiones.

$\frac{1}{3}$ FEC: En Fec $\frac{1}{3}$ el encabezado del paquete bluetooth tiene una longitud de 16 bits, pero se repite tres veces para garantizar que no haya errores en la transmisión del encabezado. Como resultado, el encabezado consume un total de 54 bits. Este enfoque requiere mucho tiempo, pero es confiable.

$\frac{2}{3}$ FEC: En Fec $\frac{2}{3}$ cada tercer bit de datos se usa para corregir errores en los dos bits anteriores. Aquí la señal sería más robusta y se puede recibir más fácilmente, con menos ruptura en la lluvia, por ejemplo. Es menos eficiente, con menos datos disponibles para transmitir la imagen.

Automatic Repeat Request (ARQ).- El ARQ se basa en el reenvío de los paquetes; el dispositivo receptor detecta errores y solicita retransmisiones. Como tal, ARQ requiere un canal de retroalimentación entre el Receptor y el Transmisor. Después de cada palabra de código, el receptor devuelve un acuse de recibo positivo (ACK) o negativo (NAK) al transmisor. Cuando el receptor detecta un error en un paquete, solicita automáticamente al transmisor que reenvíe el paquete. Este proceso se repite hasta que el paquete está libre de errores o el error continúa más allá de un número predeterminado de eventos de retransmisión. [22]

Seguridad

Bluetooth incorpora varios mecanismos de seguridad que lo convierten en uno de los protocolos más confiables frente a ataques y capturas de datos. Se definen mecanismos de seguridad en las siguientes capas de protocolo:

- Seguridad a nivel de banda base
- Seguridad a nivel de enlace

Seguridad a nivel de banda base: La seguridad a nivel de banda base en bluetooth emplea la técnica de salto de frecuencias FHSS, que consiste en dividir la banda en 79 canales de longitud 1 MHz y realizar 1600 saltos por segundo.

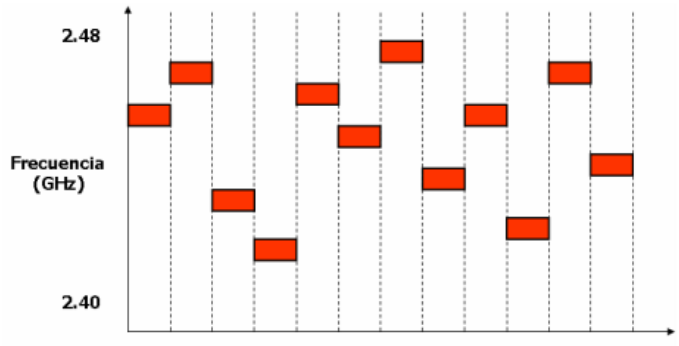


Figura 16 Técnica FHSS [25]

Durante el proceso de establecimiento de la conexión en una piconet, el dispositivo maestro genera una tabla pseudoaleatoria con la secuencia o el patrón de saltos de frecuencia que debe utilizar los dispositivos pertenecientes a la piconet durante las comunicaciones este efecto se puede visualizar en la figura 16.

Seguridad a nivel de enlace: En la seguridad a nivel de enlace se definen tres mecanismos que son:

Autentificación.- La autenticación es el proceso por el cual un dispositivo Bluetooth verifica su identidad en otro dispositivo para poder acceder a los servicios que ofrece. La autenticación no requiere la intervención del usuario; implica un esquema desafío respuesta entre cada par de dispositivos que emplea una clave de enlace secreta común de 128 bits. Consecuentemente, este esquema se utiliza para autentificar dispositivos, no usuarios.

Autorización.- La autorización es el procedimiento que determina los derechos que tiene un dispositivo bluetooth para acceder a un servicio que ofrece un sistema. El mecanismo de autorización en dispositivos bluetooth se lleva a cabo mediante niveles de confianza, los dispositivos tienen tres niveles de confianza, los cuales determinan la capacidad de acceso a los servicios: total, parcial o restringida y nula.

- Un dispositivo de confianza mantiene una relación de emparejamiento y dispone de acceso sin restricciones a todos los servicios.

- Un dispositivo de confianza restringida mantiene una relación de emparejamiento y sólo dispone de acceso restringido a uno o varios servicios, pero no a todos.

- Un dispositivo no confiable es aquel que puede o no mantener tener una relación de emparejamiento pero que no es de confianza. No se le permite el acceso a ningún servicio.

Cifrado de datos.- El cifrado de datos protege la información que se transmite en un enlace entre dispositivos bluetooth, garantiza la confidencialidad del mensaje trasmitido de forma que si un paquete es trasmitido por un usuario que no posea la clave de descifrado, el mensaje le resultara ininteligible.

Su implementación es opcional, pero necesita que se haya producido anteriormente una autenticación, el maestro y esclavo deben ponerse de acuerdo en utilizar cifrado o no; si lo hacen deben ponerse de acuerdo en el tamaño de la clave del cifrado [25].

2.2.16 Tecnología Zigbee

Zigbee es una especificación para un conjunto de protocolos de comunicación basados en el estándar IEEE 802.15.4 para redes inalámbricas de área personal. [26] Opera en la banda ISM de 2,4 GHz, permite comunicaciones inalámbricas confiables, con bajo consumo de energía y bajas tasas de transmisión para aplicaciones de monitoreo y control; específicamente útil para redes de sensores en entornos industriales, médicos y, sobre todo, demóticos [27].

Ventajas

- Ideal para conexiones punto a punto y punto multipunto

- Diseñado para el direccionamiento de información y el refrescamiento de la red
- Óptimo para redes de baja tasa de transferencia de datos

- Son más baratos y de construcción más sencilla

- Tiene un abajo nivel de radiación y por tanto se puede utilizar en el sector medico
- Baja ciclo de trabajo - Proporciona larga duración de la batería
- Reduce tiempos de espera en el envío y recepción de paquetes

Desventajas

- Tasa de trasferencia muy baja
- Solo manipula textos pequeños comparados con otras tecnologías
- Trabaja de manera que no puede ser compatible con bluetooth en todos sus aspectos: tasa de trasferencia ni la misma capacidad de soporte de nodos
- Tiene menos cobertura porque pertenece a las redes inalámbricas de área personal WPAN [27].

Tipos de dispositivos

Se definen tres tipos distintos de dispositivo Zigbee según su papel en la red como se muestra en la figura 17.

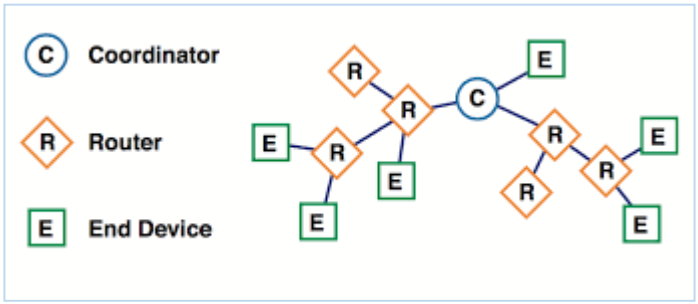


Figura 17 Dispositivos Zigbee [27]

Coordinador Zigbee.- El coordinador es el tipo de dispositivo más completo, debe existir uno por red. Sus funciones son: encargarse de controlar la red y los caminos que deben seguir los dispositivos para conectarse entre ellos.

Reuter Zigbee.- El router Zigbee interconecta dispositivos separados en la topología de la red, además de ofrecer un nivel de aplicación para la ejecución de código de usuario.

Dispositivo final.- Este dispositivo posee la funcionalidad necesaria para comunicarse con su nodo padre (el coordinador o un router), pero no puede transmitir información destinada a otros dispositivos. De esta forma, este tipo de nodo puede estar dormido la mayor parte del tiempo, aumentando la vida media de sus baterías. Un ZED tiene requerimientos mínimos de memoria y es por tanto significativamente más barato [27].

Topologías de red

En una red Zigbee pueden haber hasta 254 nodos, no obstante según la agrupación que se haga, se pueden crear hasta 255 clusters de nodos con lo cual se puede llegar a tener 64770 nodos, para lo que existe la posibilidad de utilizar varias topologías de red [28].

La tecnología Zigbee permite tres topologías de red como se muestra en la figura 18:

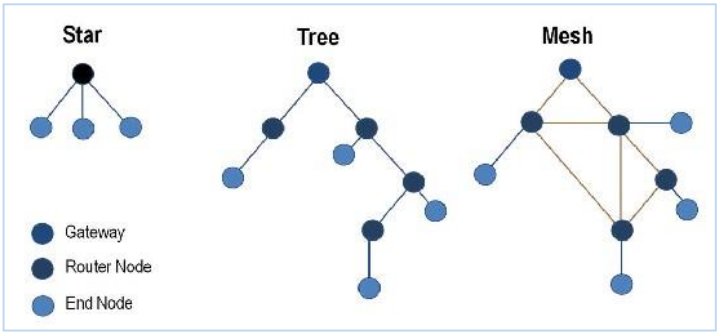


Figura 18 Topologías Zigbee [28]

- Topología en estrella: el coordinador se sitúa en el centro
- Topología en árbol: el coordinador será la raíz del árbol.
- Topología de malla: al menos uno de los nodos tendrá más de dos conexiones

Arquitectura de Protocolo

La Alianza Zigbee dirige el desarrollo de las capas superiores, por medio de la definición del perfil de aplicación. Estos perfiles hacen uso de un modelo de referencia simplificado de cinco capas de la ISO/OSI, donde se puede ver las capas que maneja el estándar 802.15.4 y las capas superiores que se le atribuyen a esta alianza como se muestra en la figura 19.

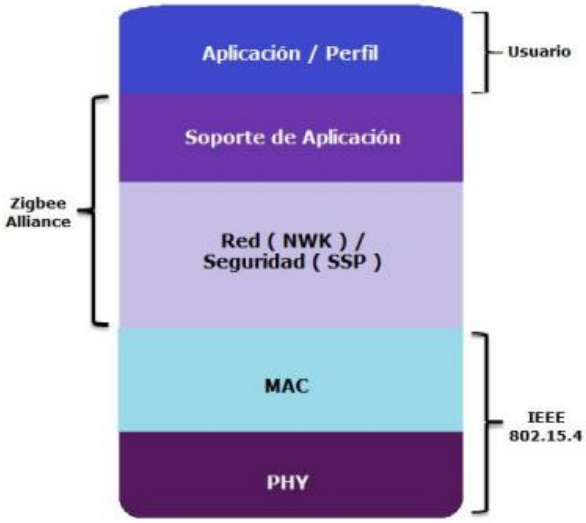


Figura 19 Pila de protocolos 802.15.4 [29]

PHY.- La capa inferior PHY provee la interfaz con el medio físico donde ocurre la comunicación, es el primer componente del modelo ISO/OSI y se encarga de:

- Control de activación y desactivación de transmisor-receptor y actuadores
- Detección de energía (dependiendo del transmisor-receptor).
- Calidad del enlace.
- Asignación de canales.
- Selección de canales.
- Medición de variables.
- Transmisión y recepción de los paquetes de mensajes a través del medio

Enlace de datos.- El enlace comprende la subcapa MAC, juntamente con la subcapa de Control de Enlace Lógico. La capa MAC proporciona control de acceso hacia el canal y confiabilidad en la entrega de datos. El estándar IEEE 802.15.4 usa el algoritmo CSMA/CA con un mecanismo que evita las colisiones de datos, el cual chequea la disponibilidad del canal antes de transmitir y así evitar colisiones con otros transmisores.

Capa de red.- La capa de red permite el correcto uso del subnivel MAC y ofrecer una interfaz adecuada para su uso por parte dela capa de aplicación. En esta capa se brindan los métodos necesarios para:

- Transmisión y recepción de los paquetes de mensajes a través del medio

- Iniciar la red
- Unirse a la red
- Enrutar paquetes dirigidos a otros nodos en la red
- Proporcionar los medios para garantizar la entrega del paquete al destinatario final, filtrar paquetes recibidos, cifrarlos y autentificarlos

Soporte de aplicación.- El soporte de aplicación es responsable de mantener el rol que el nodo juega en la red, filtrar paquetes a nivel de aplicación, mantener la relación de grupos y dispositivos con los que la aplicación interactúa y simplificar el envío de datos a los diferentes nodos de la red.

Aplicación.- La capa final (aplicación) es donde se encuentran los dispositivos Zigbee que se encargan de definir el papel del dispositivo en la red, si el actuará como coordinador, ruteador o dispositivo final [29].

Seguridad

La seguridad Zigbee, basada en un algoritmo AES de 128 bits, incrementa el nivel de seguridad proporcionado por IEEE 802.15.4. Los servicios de seguridad de Zigbee incluyen métodos para el establecimiento y el transporte de claves, la gestión de dispositivos y la protección de marcos.

Zigbee utiliza un modelo de confianza abierto donde los niveles de la pila de protocolos confían entre sí, esto es posible ya que los dispositivos Zigbee normalmente son microcontroladores inalámbricos de único chip.

- **Centro de confianza:** El centro de confianza para que una red Zigbee funcione con seguridad aceptable debe incluir un único dispositivo denominado centro de confianza (este rol normalmente lo asume el Coordinador Zigbee) que controle el acceso y distribuya las claves cuyas principales funcionalidades son:
  - Mantener y distribuir las claves de red
  - Autenticar un dispositivo en la red
  - Permite seguridad extremo a extremo entre dispositivos

- **Claves de seguridad:** Zigbee usa tres tipos de claves para administrar la seguridad: clave maestra, de red y de enlace.

    - **Claves maestras.-** Las claves maestras son opcionales y no se utilizan para encriptar marcos, sino que se usan como un secreto compartido inicial entre dos dispositivos cuando realizan el procedimiento de establecimiento de clave (SKKE, por sus siglas en inglés) para generar claves de Enlace.

Las claves que se originan en el centro de confianza se llaman claves maestras del centro de confianza, mientras que todas las demás se llaman claves maestras de la capa de aplicación.

    - **Claves de red.-** La**s** claves de red encargadas de la seguridad de la capa de red en la tecnología Zigbee. Todos los dispositivos que se encuentren en una red Zigbee comparten la misma clave. Las claves de red de alta seguridad siempre deben enviarse encriptadas y de forma inalámbrica, mientras que las claves de red de seguridad estándar pueden enviarse encriptadas o no encriptadas.

    - **Claves de enlace.-** Son claves opcionales, protegen los mensajes de difusión única entre dos dispositivos en la capa de aplicación. Las claves que se originan en el centro de confianza se llaman claves de enlace del centro de confianza, mientras que todas las demás se llaman claves de enlace de la capa de aplicación. [29]

Modos de seguridad

Modo de seguridad estándar.- En este modo la lista de dispositivos, claves maestras, claves de enlace y claves de red puede mantenerse por el centro de confianza o por los mismos dispositivos. El centro de confianza sigue siendo responsable de mantener una clave de red estándar y de controlar las políticas de admisión a la red. En este modo, los requisitos de memoria para el Centro de confianza son mínimos a comparación de los del modo de alta seguridad.

Modo de alta seguridad.- En este modo, el centro de confianza mantiene una lista de dispositivos, claves maestras, claves de enlace y claves de red que se necesita para

controlar y cumplir con las políticas de actualización de claves de red y de admisión a la red. A medida que crece la cantidad de dispositivos en la red, también crece la memoria requerida para el Centro de confianza [29].

2.2.17 Tecnología Wi-Fi

Wi-Fi es una tecnología que comprende un conjunto de estándares para redes inalámbricas basados en las especificaciones IEEE 802.11, lo cual asegura la compatibilidad e interoperabilidad en los equipos certificados bajo esta denominación. Esta tecnología permite conectar a internet equipos electrónicos mediante señales de radio frecuencia que operan en la banda libre de 2.4 y 5 GHz [30].

Ventajas

- Asignación de canales.
- Al ser redes inalámbricas, la comodidad que ofrecen es muy superior a las redes cableadas porque cualquiera que tenga acceso a la red puede conectarse desde distintos puntos dentro de un rango suficientemente amplio de espacio
- Una vez configuradas, las redes Wi-Fi permiten el acceso de múltiples ordenadores sin ningún problema ni gasto en infraestructura, ni gran cantidad de cables
- La Wi-Fi Alliance asegura que la compatibilidad entre dispositivos con la marca Wi-Fi es total, con lo que en cualquier parte del mundo podremos utilizar la tecnología Wi-Fi con una compatibilidad total

Desventajas

- Una de las desventajas que tiene el sistema Wi-Fi es una menor velocidad en comparación a una conexión cableada, debido a las interferencias y pérdidas de señal que el ambiente puede acarrear
- Esta tecnología no es compatible con otros tipos de conexiones sin cables como Bluetooth, GPRS, UMTS, etc.

- La potencia de la conexión del Wi-Fi se verá afectada por los agente físicos que se encuentran a nuestro alrededor, tales como: arboles, paredes, montañas, etc. Dichos factores afectan la potencia de compartimiento de la conexión Wi-Fi con otros dispositivos [31].

Componentes de una red Wi-Fi

Para que un ordenador pueda comunicarse de forma inalámbrica, necesita disponer de una serie de dispositivos que garanticen dicha comunicación, básicamente son 3:

Adaptador de red.- El adaptador de res es un equipo de radio con transmisor, receptor y antena que puede venir integrado en el equipo o instalado de forma independiente y que es el que permite comunicarse de forma inalámbrica.

Puntos de acceso.- Los puntos de acceso son como una estación base utilizada para gestionar las comunicaciones entre los distintos terminales de red. Los puntos de acceso funcionan de forma autónoma sin necesidad de estar conectados a ningún ordenador.

Router.- El router permite conectar un punto de acceso a internet [32].

Topologías de red

Las redes de área local inalámbrica WLAN usan la tecnología wi-fi como herramienta para poder levantar sistemas de comunicaciones, en base a esto podemos decir que existen 3 modelos lógicos de red.

Ad – Hoc.- Ad – hoc también llamadas redes entre pares, varios dispositivos conforman la red de este tipo para intercambiar información sin contar con el apoyo de elementos auxiliares APs como se muestra en la figura 20.

- Red temporal para reuniones o conferencias.
- Solo punto a punto
- Fácil de configurar.
- Conjunto de servicios básico independiente.
- Las redes independientes no utilizan Acces Point (AP).

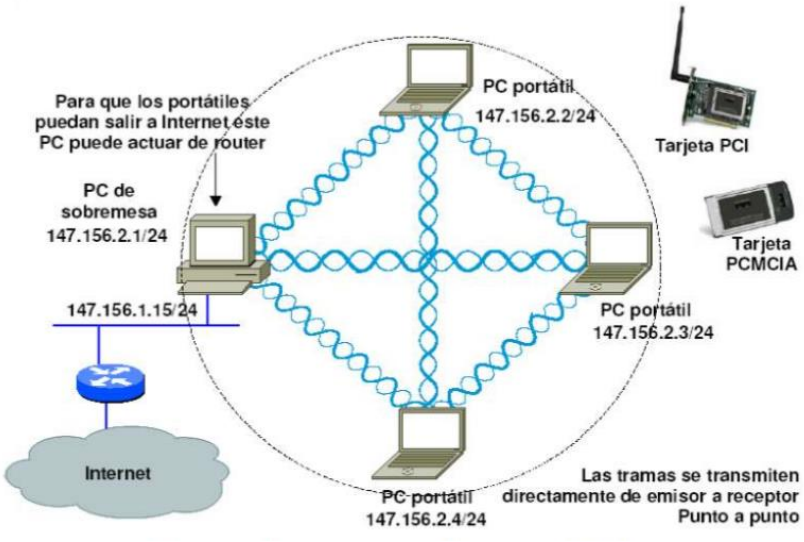


Figura 20 Estructura de una red Ad-Hoc [33]

Infraestructura.- En el modelo infraestructura los dispositivos cliente se conectan a los AP en lo que se denominan células, y pueden intercambiar información con dispositivos conectados a su mismo AP (siempre a través de éste). Por lo tanto, no tienen que encontrase en el rango de alcance para poder comunicarse. Al ser una comunicación centralizada, si se cae el AP ninguno de los dispositivos podrá comunicarse entre sí, la figura 21 muestra un esquema completo de este tipo de red.

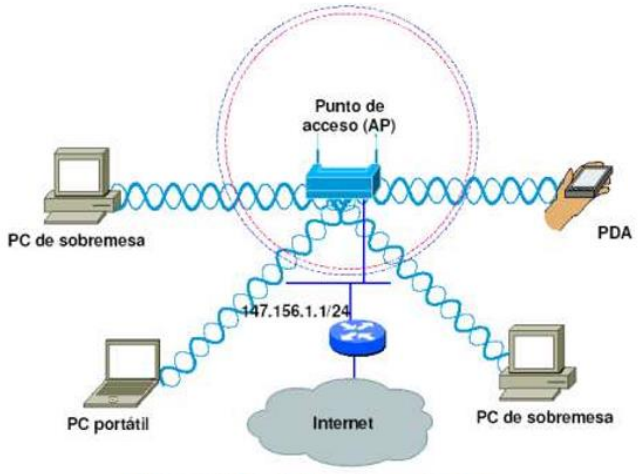


Figura 21 Estructura de una red tipo Infraestructura [33]

Mesh.- Mesh es una de red compuesta por nodos organizados en una topología malla. El área de cobertura, de todos los nodos actuando como uno sólo, se llama nube de la malla, el esquema Mesh se muestra en la figura 22 [33].

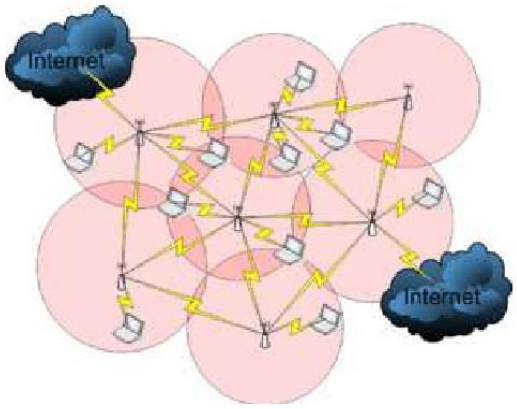


Figura 22 Estructura de una red tipo Mesh [33]

Protocolos TCP/IP

Contienen un conjunto de protocolos que se componen de dos grupos: el primero garantiza la comunicación inalámbrica entre las estaciones llamados protocolos wi-fi mientras que el otro se ocupa del intercambio de información entre los terminales TCP/IP, en la tabla 1 se muestra la pila de protocolos que rigen al estándar 802.11.

Tabla 1 Protocolos TCP/IP [31]

Pila de protocolos TCP/IP	
5	**Aplicación**
4	**Trasporte**
3	**Internet**
2	**Enlace de datos**
1	**Física**

Los dispositivos Wi-Fi deben escoger ciertos parámetros antes de poder establecer la comunicación. Estos parámetros deben configurarse adecuadamente para poder establecer conectividad a nivel de la capa uno.

- Canal de radio
- Modo de operación del radio
- Nombre de la red
- Tipo de seguridad

Wi-Fi ofrece una conexión local. No provee la funcionalidad de enrutamiento (encaminamiento, ruteo), la cual es suministrada por los protocolos de las capas superiores. [33]

Seguridad

En cuanto a la seguridad la tecnología Wi-Fi no se libra de posibles ataques y riesgos a la información por parte de los usuarios y dispositivos que utilizan esta tecnología; de hecho son foco de numerosos ataques. El primer paso para asegurar una red WLAN, es conocer cuáles son los ataques que este tipo de redes pueden sufrir. Éstos pueden ser divididos en dos grandes grupos:

Ataques Pasivos.- En los ataques pasivos el principal objetivo del atacante es obtener información. Los ataques mencionados anteriormente suponen un primer paso para ataques posteriores. Algunos ejemplos de este tipo de ataques serían el espionaje, escuchas, wardriving y los ataques para el descubrimiento de contraseñas.

Ataques Activos.- Estos ataques implican la modificación en el flujo de datos o la creación de falsos flujos en la transmisión de datos. Pueden tener dos objetivos diferentes: pretender ser alguien que en realidad no se es o colapsar los servicios que puede prestar la red. Algunos ejemplos de este tipo de ataques son el spoofing, la instalación de puntos de acceso no autorizados o Rogue APs, el secuestro de sesiones (Hijacking) y la denegación de servicio (DOS). El segundo paso es el conocimiento de los mecanismos de seguridad aplicables en redes WLAN, los cuales son:

- **WEP (Wired Equivalent Privacy):** WEP fue el primer mecanismo de seguridad que se implementó bajo el estándar de redes inalámbricas IEEE 802.11 para cifrar los datos que se transfieren a través de una red inalámbrica. Es un mecanismo de seguridad básico del que han sido demostradas numerosas vulnerabilidades.

- **WPA (Wi-Fi Protected Access):** WPA es un Estándar desarrollado por la Wi-Fi Alliance, basado en un borrador del estándar IEEE 802.11i, para mejorar el nivel de cifrado existente en WEP e incorporar además un método de autenticación.

- **IEEE 802.11i:** El 802.11i es un estándar que define el cifrado y la autentificación para complementar, completar y mejorar la seguridad en redes WLAN proporcionada por WEP

- **WPA2 (Wi-Fi Protected Access v2):** Es la implementación aprobada por Wi-Fi Alliance interoperable con IEEE 802.11i. El grupo WPA2 de Wi-Fi Alliance es el grupo de certificación del estándar IEEE 802.11i, para lo cual se basa en las condiciones obligatorias del mismo. En función de la configuración de un sistema WPA2 su comportamiento será similar a la de un sistema WPA o un sistema IEEE 802.11i.

El funcionamiento básico de estos mecanismos se basa en el cifrado de la información de usuario en el interfaz aire (entre el terminal de usuario y el punto de acceso WLAN). Además, todos excepto WEP, implican autenticación de usuario. En el caso del mecanismo WEP la única autenticación que se realiza es la autenticación de terminal, pero no contempla ningún otro modo de autenticación de usuarios ni de punto de acceso.

Adicionalmente, es posible emplear en redes WLAN soluciones de seguridad ya empleadas en otros tipos de redes (cableada o inalámbrica). Así, por ejemplo, mecanismos de seguridad como SSH, HTTPS, SSL, IPSec VPN, PPTP VPN y L2TP VPN son aplicables no sólo a redes WLAN sino también a otro tipos de redes [34].

2.2.18 Tecnología Z-Wave

Z-wave es una tecnología de comunicación inalámbrica sin estandarizar. Sin embargo, existe la Alianza Z-wave para garantizar la interoperabilidad de los dispositivos empleados. El estándar es cerrado y por tanto es necesario ser miembro para acceder a él. Trabaja en la banda de los 868 MHz evitando la gran cantidad de emisoras en la banda de los 2,4GHZ y puede llegar a trabajar a 40 kbit/s pudiendo operar en rangos de hasta 30 metros en condiciones ideales [35].

Ventajas

- Domotizar con tecnología inalámbrica Z-Wave permite eliminar el uso de cables, además los módulos se comunican entre ellos de forma bidireccional formando una red mallada.

- Asociación directa entre módulos hace que se pueda prescindir del Controlador central.

- Evita crear nuevas escenas en el Controlador, con lo cual le quitamos trabajo y tenemos una latencia más baja para el resto de tareas.

Desventajas

- Solo admiten un parámetro en la acción que lo desencadena [36].

Tipos de Dispositivos Z wave

En la tecnología Z wave se definen dos tipos básicos de dispositivos que son:

Controladores.- Controladores son aquellos que inician y envían los comandos de control necesarios a los diferentes nodos. Los controladores siempre son conocedores de la organización de toda la red con el objeto de poderse comunicar con cualquier nodo.

Esclavos.- Esclavos son aquellos que obedecen, ejecutan y responden a las órdenes de los controladores. Los esclavos son los dispositivos que reciben comandos, los ejecutan y responden. Un esclavo no puede intercambiar información directamente con otro esclavo [35].

Topología de red

La topología de red es tipo malla y cada elemento se comporta como un nodo que puede ser receptor o emisor reenviando el mensaje. Permite también realizar agrupaciones para asociar la misma funcionalidad a todos los elementos del grupo. El principal inconveniente es el elevado consumo eléctrico que lleva asociado ya que el hardware encargado de la comunicación está permanentemente activo, la figura 23 muestra el esquema de red antes mencionado [35].

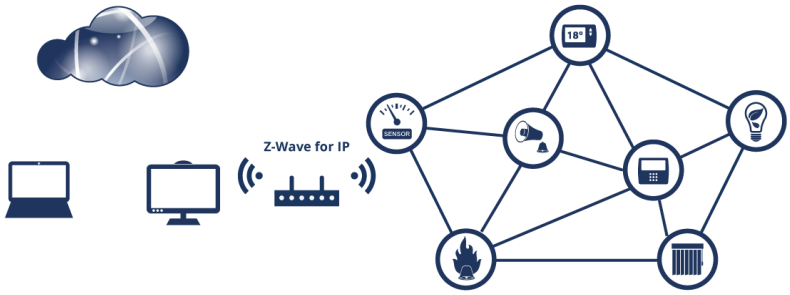


Figura 23 Estructura de una red tipo Mesh Z-wave [35]

Seguridad

Actualmente para que un dispositivo reciba la certificación Z-wave es obligatorio que todos los dispositivos implementen el framework de seguridad avanzada Z-Wave S2. Este sistema provee a la tecnología Z-Wave nuevos niveles de impenetrabilidad.

Este mecanismo está basado en tres clases de seguridad cada una con una clave de red distinta. Cada una de estas clases dictan la manera en la que un nodo es incluido en la red y por tanto su funcionamiento posterior.

S2 Access Control.- S2 Access es la clase más segura entendida para incluir dispositivos como seguros de apertura de puerta o aperturas de garaje.

S2 Authenticated.- S2 Authenticated está pensada para dispositivos comunes como sensores o interruptores de luz.

S2 Unautenticated.- S2 Unautenticated es la menos segura y está pensada para dispositivos tan sencillos que no sean capaces de implementar un sistema de autenticación mínimo cuando se unen a la red Z-Wave. Un ejemplo podría ser un llavero que controla un par de lámparas en una casa de campo.

S0 compatible.- S0 es una clase que contiene una clave para compatibilidad con dispositivos antiguos que no implementen S2. Los nodos S2 transportan entre ellos la clave S0 como S2 Unautenticated; usando la clave ECDH para el intercambio de claves en lugar del intercambio de claves S0 [37].

En la tabla 2 se indica un resumen de las tecnologías comunes al IoT descritas anteriormente.

Tabla 2 Resumen de protocolos comunes en el IoT [25] [20] [30]

Parámetro\Tecnología	Bluetooth Clásico	Wifi	Zigbee	Z Wave
Estándar	802.15.1	802.11.a/g/n	802.15.4	Alianza Z wave
Frecuencia	2,4 GHz	2,4 GHz	2,4 y 5 GHz	415–815 MHz
Tasa Máxima de Bits	1 - 3 Mbps	2 Mbps	0,25 Mbps	
Tipo de datos	Audio, imagen, datos	Audio, video, datos	Pequeños paquetes de datos	Pequeños paquetes de datos
Rango Máximo Cobertura	100 m	50 m -30 Km	100 m	5 m
Vida de la Batería	Días	Alimentada por la red	100 días	Días
Tamaña de red	7 nodos	Depende el tipo de red	64.000+ nodos	Nodos +
Potencia típica de transmisión	100 mW	200mW - 4 W	1 - 10 mW	220mA
Ancho de canal	1 MHz	22 MHz	2 MHz	
Modulación	GFSK	OFDM	BPSK	
Sensibilidad del receptor	-90 dBm	-87 a -93 dBm	-85 dBm	
Número de canales	79 canales	11 canales	16 canales	
Difusión del espectro	FHSS	FHSS	DSSS/CSS	
Método de acceso al medio (MAC)	TDMA	CSMA	CSMA/CA, TDMA	
Identificación MAC	48 bits	8 y 32 bits	16 y 64 bits	
Control de errores FEC/CRC	8 bits con ACK opcional	CRC de 24 bits y ACK	16 bits con ACK opcional	
QoS	Si	Si	Si	
Latencia	< 100 ms	< 3 ms	< 5 ms	
Topología de red	Estrella	Had-hoc Infraestructura, Mesh	Estrella, Cluster, Malla	
Tiempo de unión a red	< 10 s	< 1 s	< 1 s	
Consumo de energía	Medio	Alto	Muy bajo	
Complejidad del protocolo	Sencillo	Complejo	Sencillo	
Seguridad	Autenticación, Encriptación	Autenticación, Encriptación	Autenticación, Encriptación	
Aplicación	WPAN	WLAN +	LR-WPAN	
Fortalezas	Bajo Costo y múltiples perfiles de aplicación	Costo medio y múltiples perfiles de aplicación	Fiabilidad, bajo consumo y bajo costo	

Bandas sin licencia en redes inalámbricas

Las bandas sin licencia permiten la operación de dispositivos de radiocomunicaciones sin una planificación centralizada por parte de la Autoridad de Comunicaciones, es

decir, sin una autorización individual de cada estación tal que asegure la asignación de una frecuencia o canal para uso exclusivo de la misma. La banda se destina íntegramente a tales dispositivos, sin subdivisión de canales, estableciéndose ciertos requerimientos básicos de convivencia, tales como límites de potencia o de densidad de potencia radiadas, anchura de banda mínima, etc. La coordinación corre por cuenta de los usuarios, pero se apoya principalmente en la inmunidad contra interferencias, propia de la tecnología empleada, y el modo de acceso múltiple a la banda. A continuación se muestra algunas características de las tres bandas de frecuencia libres (900 MHz, 2.4 y 5 GHz) como indica la tabla 3.

- Las Frecuencias no-licenciadas son una plataforma amplia, barata y rápida para construir soluciones inalámbricas.

- A pesar del gran uso del espectro de licencia libre, sigue siendo una excelente plataforma para construir enlaces inalámbricos de bajo precio, rápidos y confiables. [38]

Tabla 3 Bandas sin licencia [28]

BANDAS NO LICENCIADAS			
Parámetro	**900 MHz**	**2.4 GHz**	**5 GHz**
Popularidad	No usadas ampliamente	Ampliamente usadas	Ampliamente usadas
Velocidad	Bajo rendimiento	Alto rendimiento	Alto rendimiento
Costo	No caro	No caro	No caro
Frecuencia	Abarrotado	Abarrotado	No abarrotado
Alcance	Alcance débil	Alcance promedio	Alcance promedio
Aplicación	Mesh, ptmp cortos con muchos obstáculos	Mesh, ptp, ptmp	Backhaul, ptp, ptmp

2.2.19 Descripción del periférico utilizado previo a un análisis comparativo descrito en la tabla 4.

Se presenta un análisis comparativo de equipos compatibles con SDR entre: USRP, HackRF One y Ubertooth, seleccionado a HackRF one por tener ventajas en costo,

compatibilidad con las tres tecnologías analizadas (Bluetooth, wifi, Z wave), rango de frecuencia superior a 2.4 GHz, etc. A continuación se describe el periférico.

HackRF One es un periférico Software Defined Radio capaz de transmitir o recibir señales de radio desde 1 MHz hasta 6 GHz fabricado por Great Scott Gadgets. Fue diseñado para facilitar el desarrollo y testeo (tanto auditoría como hacking) de tecnologías de comunicación radio tanto actuales (soporta LTE), como en desarrollo para las nuevas generaciones de tecnologías radio y sus correspondientes protocolos. HackRF One que se muestra en la figura 24 es una plataforma de hardware libre que puede ser usada como un periférico vía USB, o programada para operar de forma autónoma.

- ✓ **Características**

  - Tiene un rango de operación en frecuencia desde 1 MHz hasta 6 GHz.
  - Es un transceiver con capacidad de operación half-duplex.
  - Tiene una capacidad de muestreo de hasta 20 millones de muestras por segundo, pudiéndose alcanzar las 21,5 en función del tipo de controlador USB 2.0 HS que incluya el computador al que se conecta.
  - Es compatible con los principales programas para SDR tanto en Windows como en Linux, lo que incluye tanto a SDR# como a GNU Radio.
  - Puede configurar vía software los amplificadores de ganancia, con 3 etapas dedicadas para recepción y 2 etapas para transmisión.
  - Puede configurar vía software los filtros de señal en banda base, con un máximo de ancho de banda de señal de 28 Mhz, y con una caída de 3 dB hasta 30 MHz.
  - Permite controlar vía software la potencia suministrada al puerto de la antena, con hasta 50 mA a 3.3 V.
  - Conector de antena SMA hembra.

- Un conector SMA hembra para sincronizar el reloj, tanto a la entrada como a la salida, lo que permite conectar varios HackRF One y hacerlos trabajar conjuntamente, reduciendo los problemas de jitter.

- Botones para configurar convenientemente el dispositivo.

- Cabeceras de pines internos para una posible expansión de la placa usando shields.
- Uso de la interfaz USB 2.0 High Speed.

- Todo el sistema se alimenta a través de la conexión USB, sin necesidad de añadir una fuente de alimentación externa, la que le confiere una gran portabilidad.

- Es una plataforma de hardware libre.

Figura 24 Equipo HackRF One [39]

La selección del dispositivo se realizó en base a la capacidad de análisis de señales de radio que este posee y la operación de la antena; en el desarrollo de aplicaciones con SDR para un entorno IoT la recepción de la antena debe tener como mínimo los 2.4 GHz, el equipo opera en un rango amplio de frecuencias integrando la anterior mencionada.

2.2.20 Componentes del equipo: HackRF One

Figura 25 Antena ANT500 [39]

Antena ANT500.- Antena telescópica que tiene una longitud variable de 20 a 88 cm como se muestra en la figura 25, opera entre 75 MHz y 1 GHz, usa un conector SMA macho y está diseñada para tener una resistencia de antena de 50 Ohmios, de forma que pueda ser adaptada a una pista con impedancia característica de 50 Ohmios como carga minimizando las pérdidas como se muestra en la figura 26.

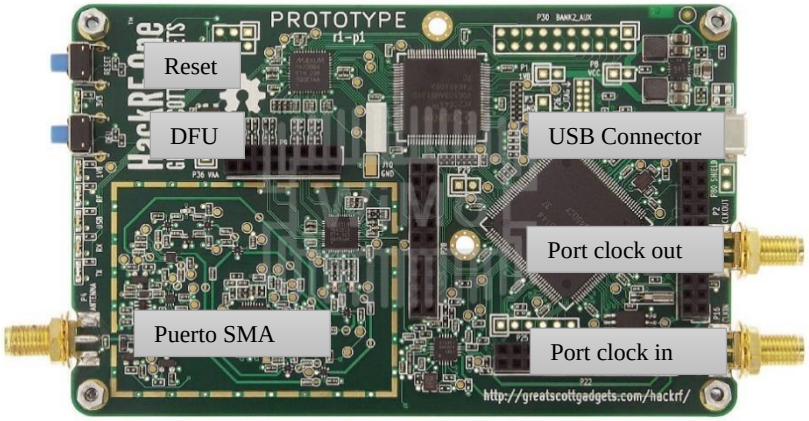


Figura 26 Placa HackRF One [39]

Puerto SMA.- Puerto de conexión para la antena ANT500 o cualquier distribución que sea compatible con este puerto, algunas características son:

- Impedancia: 50 Ohm
- Frecuencia: 0 ~ 6 GHZ
- Material del conector: Copper
- Longitud: 13 mm
- Longitud de la puntada: 4 mm

Conector USB.- Conector que sirve como fuente de alimentación del periférico que va conectado al computador: USB 2.0 High Speed.

Reset y DFU.- Sockets que permiten el reseteo del equipo así como la actualización del firmware respectivamente. La segunda acción no es necesaria a menos que el software de configuración este con algún fallo.

Clock in – Clock out.- Puertos que sirven para la sincronización de reloj cuando se conecte diferentes dispositivos en cascada. Esto se logra por medio de un cable adaptador SMA, donde la salida del uno será la entrada del otro [39].

2.3 Propuesta de Solución

Hacking ético mediante radio definido por software que permita analizar el comportamiento de las señales por medio del desarrollo de sistemas de comunicaciones dentro de un entorno de protocolos inalámbricos para determinar vulnerabilidades y fallos en la seguridad en tecnologías comunes con sistemas IoT.

CAPÍTULO III

METODOLOGÍA

3.1 Modalidad de Investigación

El proyecto de investigación se lo realizó bajo la modalidad experimental debido a que el uso de sistemas de comunicaciones basados en SDR permitió analizar la información y el nivel de seguridad en aplicaciones IoT una vez realizado el proceso de hacking ético.

Además la información se obtuvo mediante investigación bibliográfica documental, a través de libros, proyectos de titulación, proyectos de investigación, revistas, artículos científicos, que nos permitieron profundizar en el tema y adquirir el conocimiento necesario para el desarrollo del proyecto.

3.2 Recolección de Información

La recolección de información en su mayoría se obtuvo de revistas científicas, artículos científicos, publicaciones, artículos técnicos y proyectos desarrollados en otros países relacionados a hacking ético y seguridad en el IoT.

3.3 Procesamiento y Análisis de Datos

- Recolección de información mediante libros, artículos, tesis, etc.
- Revisión de la información con el objetivo de encontrar la información necesaria para la realización del proyecto
- Lectura de artículos relacionados con el tema de investigación que muestren estrategias en busca de resultados de solución
- Análisis y revisión de resultados que permitirán detectar los errores, omisiones y eliminar respuestas que sean contradictorias y poder así discriminar la información hasta llegar a tener resultados claros de la investigación

3.4 Desarrollo del Proyecto

Para el desarrollo del proyecto se llevaron a cabo los siguientes pasos:

- Análisis de información sobre tecnologías inalámbricas aplicadas a un entorno IoT.
- Estudio de protocolos de radio frecuencia y bandas sin licencia comunes en los sistemas IoT.
- Análisis comparativo del dispositivo de irrupción de seguridad como elemento principal dentro del caso de estudio.
- Selección del software compatible con SDR que contenga aplicaciones que brinden todas las facilidades de lectura y escritura en procesamiento digital de señales.
- Diseño de réplica de señal mediante SDR usando antena FM
- Diseño de aplicaciones relacionadas a sistemas IoT con SDR
- Adecuar un entorno que contenga todos los dispositivos con los diferentes protocolos RF usados en el IoT.
- Corrección del Sistema con miras a obtener los resultados esperados
- Realizar un informe final

CAPÍTULO IV

DESARROLLO DE LA PROPUESTA

4.1 Descripción de la propuesta

El trabajo de investigación realizado presenta un análisis de señales de radio frecuencia en la banda de 2.4 GHz; señales que están relacionadas a aplicaciones IoT. El propósito es determinar vulnerabilidades en la seguridad de la información de estas aplicaciones para eso la señal de las aplicaciones debe ser detectada; se empieza el proceso de hacking desarrollando diagramas de flujo orientados a los protocolos de estudio (Bluetooth, Wi-Fi, Z wave) cada una de ellos detalla de manera simple la captación de señal, el procesamiento de información con la ayuda de herramientas auxiliares y el análisis de información a fondo. Estas aplicaciones se realizaran mediante SDR como parte de software y como parte de hardware equipos periféricos compatibles con el software utilizado, creando sistemas de comunicaciones donde la mayoría de sus elementos se basan en software. El análisis de la información captada en la fase uno será importante debido a que se verificara los fallos de seguridad informática en las tecnologías orientadas al IoT, por lo que el trabajo de investigación servirá para alertar a personas naturales o jurídicas de posibles ataques que incurrirán en riesgos no solo económicos sino también de carácter social.

El desarrollo de hacking ético al IoT requiere un estudio amplio de protocolos inalámbricos en la banda de frecuencia de 2.4 GHz; el objetivo principal es obtener la información la señal de una determinada aplicación. En el diagrama de bloques de la figura 29 se propone un proceso que permite alcanzar el objetivo que consta de 3 fases que son:

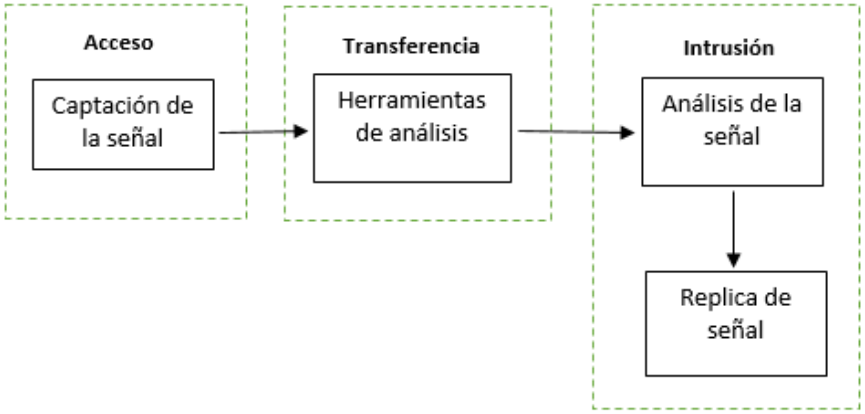


Figura 27 Diagrama de bloques: Fases de hacking

Fuente: Investigador

- **Fase de Acceso.-** Fase de acceso se recepta la señal inalámbrica de las aplicaciones desarrolladas para poder analizarlas.
- **Fase de Transferencia.-** Fase de transferencia intervienen herramientas auxiliares que receptan la información obtenida en formato binario para posteriormente analizarla.
- **Fase de Intrusión.-** Fase de intrusión se muestra toda la información (trama) de la señal en cuestión.
- **Fase de ejecución.-** En la ejecución se aplica la réplica de la señal en caso de ser posible y se realizan cambios de ser necesarios

4.2 Requerimientos para el desarrollo de hacking ético al IoT mediante SDR

Para la implementación de un hacking ético al IoT mediante SDR se necesita una serie de componentes tanto en hardware y software detallados a continuación:

4.2.1 Requerimientos de hardware

Para el desarrollo de hacking ético al IoT se necesita los siguientes requerimientos en hardware:

- Periférico captador de señal compatible con SDR
- Antena en el rango de 315MHz – 2.5GHz

Tabla 4 Comparativa entre periféricos (transceiver)

Parámetro/Tecnología	**Usrp**	**Hack rf one**	**Ubertooth**
Frecuencia de operación	DC - 6 GHz	✓ 1MHz – 6GHz	2.4 GHz
Modo de trasmisión	Full dúplex	✓ Half dúplex	Half dúplex
Frecuencia d muestreo	50MHz	✓ 20MHz	20MHz
Bits en cuadratura	Muestras de 16 bits en cuadratura	✓ Muestras de 8 bits en cuadratura	
Compatibilidad	GNU Radio, SDR#, labview, Matlab.	✓ GNU Radio, SDR#, +	GNU Radio, SDR#, +
Conector de antena	SMA	✓ SMA	Cortex Debug
Tipo de sincronización	Reloj Sincronizable input/output	✓ Reloj Sincronizable input/output	Interna
Modo de Alimentación	USB	✓ USB	USB
Hardware	Código abierto	✓ Código abierto	Código abierto
Costo	Alto	✓ Moderado	Bajo
Compatibilidad con tecnologías inalámbricas	Total	✓ Total	Solo Bluetooth

Fuente: Investigador

Para el desarrollo de sistemas de comunicaciones con SDR existen una gran cantidad de periféricos transceptores que permiten tratar la señal inalámbrica para crear aplicaciones con un nivel superior en sincronización, tomando en cuenta las necesidades del proyecto de investigación: rangos de frecuencia, compatibilidad con SDR y tecnologías aliadas al IoT, costo, etc. Se escoge el periférico *HACK RF ONE* como indica la tabla 4.

4.2.2 Requerimientos en software

El desarrollo de sistemas de comunicaciones con SDR involucra el uso de herramientas gráficas para crear diagramas de flujo de señal y generar código fuente de flujo de gráfico. No existe un software en particular que esté ligado al desarrollo del proyecto, el uso de herramientas como filtros, variables de frecuencia, moduladores, demoduladores y bloques que sirvan para tratar a la señal son factores que permiten elegir el software adecuado. La selección del software una vez conocidos los requerimientos del proyecto (herramientas de tratamiento de señal) se muestra en la tabla 5.

Tabla 5 Comparativa entre softwares de diseño – SDR

Parámetro/Software	GNU Radio Comp.	Matlab	Labview
Genero	✓ Diseño de gráficos de flujo de señal y generar código fuente de flujo de gráfico.	Software Matemático	Diseño de sistemas, Adquisición de datos, Instrumentación y Control instrumental, Procesamiento de señales, Sistemas de control industrial, Diseño de sistemas embebidos, Comunicaciones
Sistema Operativo	✓ Microsoft Windows, Mac OS X, GNU/Linux	Microsoft Windows, Mac OS X, GNU/Linux	Microsoft Windows, Mac OS X, GNU/Linux
Licencia	✓ Libre	Propietario	Propietario
Lenguaje	✓ Inglés	Inglés	Inglés
Escritorio	✓ Entorno de escritorio iterativo freedesktop	Entorno de escritorio iterativo	Express VIS
Herramientas	✓ Fuentes, bloques de procesamiento de señal, sumideros.	Diseñar curvas, clasificar datos, analizar señales, ajustar sistemas de control.	Herramientas gráficas y textuales para el procesado digital de señales. Visualización y manejo de gráficas con datos dinámicos. Adquisición y tratamiento de imágenes.
Compatibilidad	✓ Usrp, hack rf,+	Usrp, hack rf,+	Usrp, hack rf,+
Interfaz	✓ Python, SQL	C/C++, Java®, .NET, Python, SQL, Hadoop y Microsoft Excel	C/C++, Java®, .NET, Python, SQL, Hadoop y Microsoft Excel

El software seleccionado es GNU Radio Companion, su gran número de herramientas en el procesamiento de señales y la facilidad al momento de su uso garantiza el desarrollo de sistemas de comunicaciones ligados al internet de las cosas.

Comparación entre sistemas operativos de hacking ético y pruebas de penetración

Dentro de las características necesarias para elegir un sistema operativo que contenga herramientas de hacking ético y pruebas de penetración, se encuentran:

- Compatibilidad con el hardware y software seleccionado en el paso anterior
- Compatibilidad con otros paquetes que manejan herramientas de Hacking ético.
- Capacidad de trabajar con tecnologías aliadas al IoT

El sistema operativo debe ser capaz de funcionar de manera óptima con el equipo principal de trabajo, es necesario que sea lo más funcional posible debido al uso de señales en tiempo real, para ello la concordancia entre estas dos herramientas tanto en hardware y software será fundamental para poder obtener éxito en el trabajo realizado, a continuación en la tabla 6 se presenta un análisis de los sistemas operativos más utilizados para este proceso:

Tabla 6 Comparativa entre sistemas operativos libres

	Ubuntu	Kali Linux	Parrot Security	DEFT Linux
Sistema Operativo	✓ Linux	Linux	Linux	Linux
Software	✓ Código Abierto	Código Abierto	Código Abierto	Código Abierto
Basado en	✓ Debian	Debian (Testing)	Debian	Debian
Arquitectura	✓ x86, x86-64, ARM1	armel, armhf, i386, x86_64	i386, x86_64, ARM	i486
Escritorio	✓ GNOME	GNOME	MATE	LXDE
Paquetes	✓ dpkg + APT	PGP	PGP	PGP
Personalizable	✓ Si	Si	Si	Si
Lenguaje	✓ Multi-lenguaje	Multi-Lenguaje	Multi-Lenguaje	Multi-Lenguaje
Categoría	✓ Analizadores, Bluetooth, Cracker/fuerza bruta, Bases de datos, Exploit, Footprints, Forenses, Forging, Fuzzers, MITM, Wireless, SIP/VoIP	Recuperación de datos, Forensics, Auditor de seguridad, Auditor de seguridad wireless, Raspberry Pi, Seguridad, Anonimato	Seguridad, Forensics, Anonimato	Análisis de Blackberry, Android, iPhone, así como información sobre las típicas bases de datos de dispositivos móviles en SQLite utilizadas por las aplicaciones.
Estado	✓ Activo	Activo	Activo	Activo
Característica	✓ Auditor de seguridad Forense	Auditor de seguridad Forense	Forense	Forense
Herramientas de Hacking Ético	✓ 350 herramientas de testeo	300 herramientas de testeo	250 herramientas de testeo	150 herramientas de testeo
	✓ Es una distribución de Linux basada en la arquitectura de Debian. Actualmente corre en computadores de escritorio y servidores, en arquitecturas Intel, AMD y ARM.	Kali Linux, distribución basada en Debian con una colección de herramientas de seguridad y forenses. Su mayor ventaja es el soporte para la arquitectura ARM.	Parrot Security OS basada en Debian diseñada para hacking ético, pruebas informáticas forense, hacking ético, la criptografía, etc.	DEFT es una distribución basada en GNU/Linux y DART (Digital Advanced Response Toolkit), es una suite dedicada a actividades de forense digital e inteligencia.

Fuente: Investigador

Conforme al análisis de características, se optó por utilizar el sistema operativo Ubuntu, cumpliendo con los requerimientos básicos necesarios para la utilización del sistema operativo. La columna de color azul indica dichos requisitos.

Tabla de resumen en base a los requerimientos de hardware, software y sistema operativo en el desarrollo de hacking ético al IoT mediante SDR.

Tabla 7 Resumen de requerimientos para Hacking al IT [16] [2] [6]

Hardware: HackRF One	Software: Gnu Radio Companion	Sistema Operativo: Ubuntu
✓ 1MHz – 6GHz	✓ Diseño de gráficos de flujo de señal y generar código fuente de flujo de gráfico.	✓ Linux
✓ Half dúplex	✓ Microsoft Windows, Mac OS X, GNU/Linux	✓ Código Abierto
✓ 20MHz	✓ Libre	✓ Debian
✓ Muestras de 8 bits en cuadratura	✓ Inglés	✓ x86, x86-64, ARM1
✓ GNU Radio, SDR#, +	✓ Entorno de escritorio iterativo freedesktop	✓ GNOME
✓ SMA	✓ Fuentes, bloques de procesamiento de señal, sumideros.	✓ dpkg + APT
✓ Reloj Sincronizable input/output	✓ Usrp, hack rf,+	✓ Si
✓ USB	✓ Python, SQL	✓ Multi-lenguaje
✓ Código abierto		✓ Analizadores, Bluetooth, Cracker/fuerza bruta, Bases de datos, Exploit, Footprints, Forenses, Forging, Fuzzers, MITM, Wireless, SIP/VoIP
✓ Moderado		✓ Activo
✓ Total		✓ Auditor de seguridad Forense
		✓ 350 herramientas de testeo
		✓ Es una distribución de Linux basada en la arquitectura de Debian. Actualmente corre en computadores de escritorio y servidores, en arquitecturas Intel, AMD y ARM.

4.3 Manejo de GNU Radio Companion

Para el desarrollo de cualquier diagrama de flujo en GRC (GNU Radio Companion), se busca la pestaña GRC en el conjunto de aplicaciones instaladas en el sistema

operativo (Ubuntu) al ingresar muestra la pantalla principal de diseño como indica la figura 28.

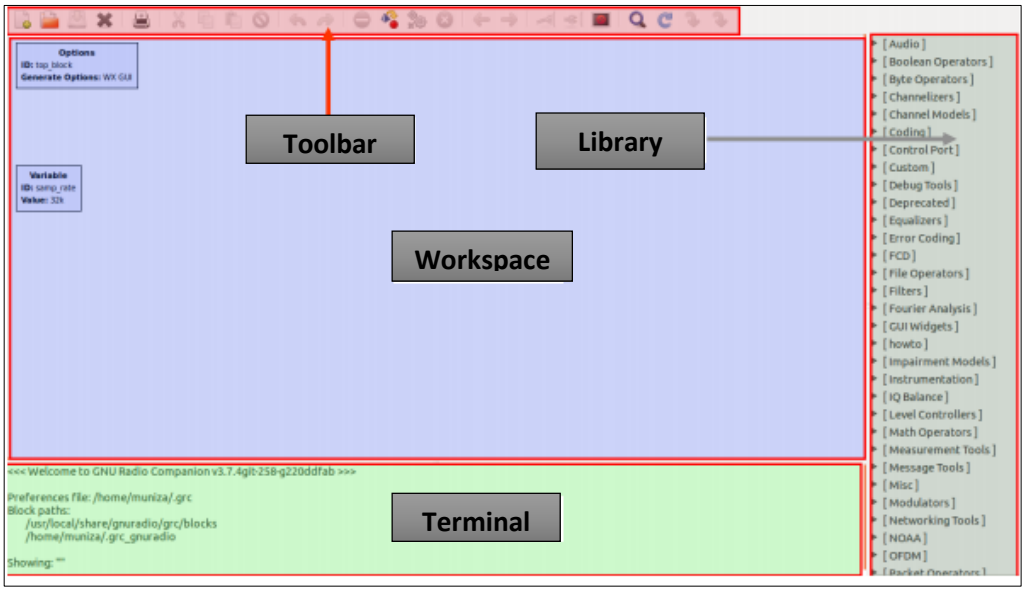


Figura 28 Interfaz GNU Radio Companion

Fuente: Investigador

En esta pantalla se realizan los diseños de los flujogramas y como se ve, la figura 28 consta de cuatro partes: “Library”, “Toolbar”, “Terminal” y “Workspace”. En “Library”, se tiene un listado de los bloques instalados y disponibles en el GRC ordenados por categorías. En “Terminal” es donde aparecen los avisos de error que existen en el grafo. El “Workspace” es la zona de diseño, donde se construye el flujograma, y por defecto contiene los bloques “Options” y “Variable”. El primero se utiliza para ajustar algunos parámetros generales del diagrama de flujo, y el segundo, se utiliza para ajustar la frecuencia de muestreo.

FASE DE ACCESO

4.4 Diseño de flujogramas en GRC (Gnu Radio Companion).

4.4.1 Estructura de flujograma como parte del proceso de hacking Bluetooth

En el apartado de librerías para poder buscarlas de manera más rápida se ubica la barra de herramientas (Toolbar) y se busca  , donde se escribe el nombre de la librería que se vaya a utilizar.

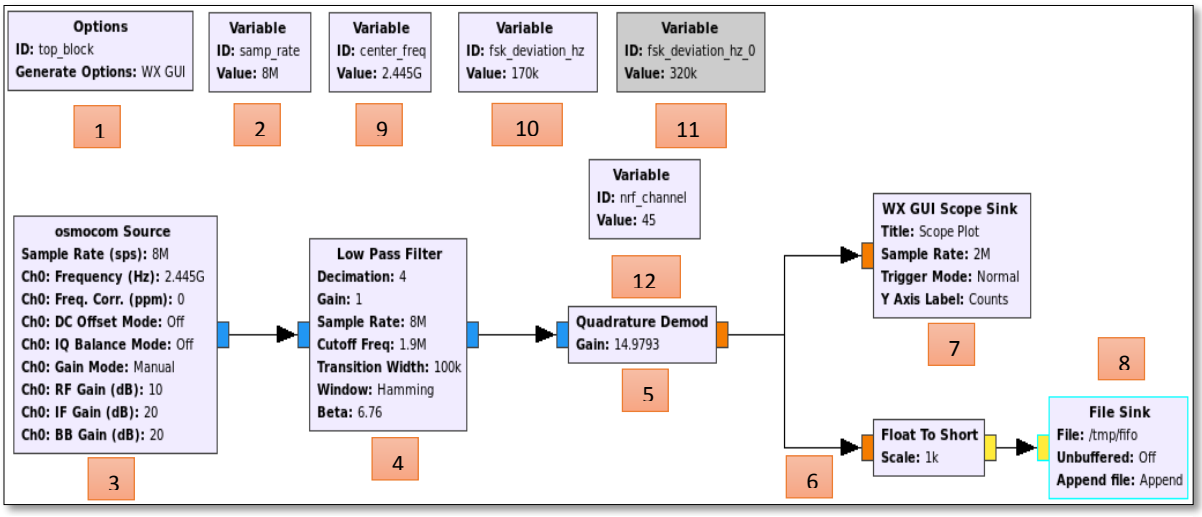


Figura 29 Diagrama de flujo para captación de señal bluetooth

Fuente: Investigador

La figura 29 representa un sistema de comunicación por radio software, el flujograma es realizado en la herramienta Gnu companion y es capaz de receptar la señal cuando exista la petición cliente – servidor. Está compuesto por dos bloques estructurados el primero tiene: fuente, filtros, demoduladores, tipo de señal a presentar, el segundo son todas las variables que contienen características propias de la señal como: frecuencia de muestreo frecuencia central, etc.

Cada uno de los flujogramas en las tres tecnologías están representados por bloques, los mismos que tienen números de identificación de acuerdo al proceso de diseño, a continuación de los diagramas se presenta una análisis de cada uno de ellos y los parámetros utilizados según el tipo de tecnología.

1.- Options:- Options muestra las opciones de generación de gráfica en función del tiempo o en función de la frecuencia como se muestra en la figura 30.

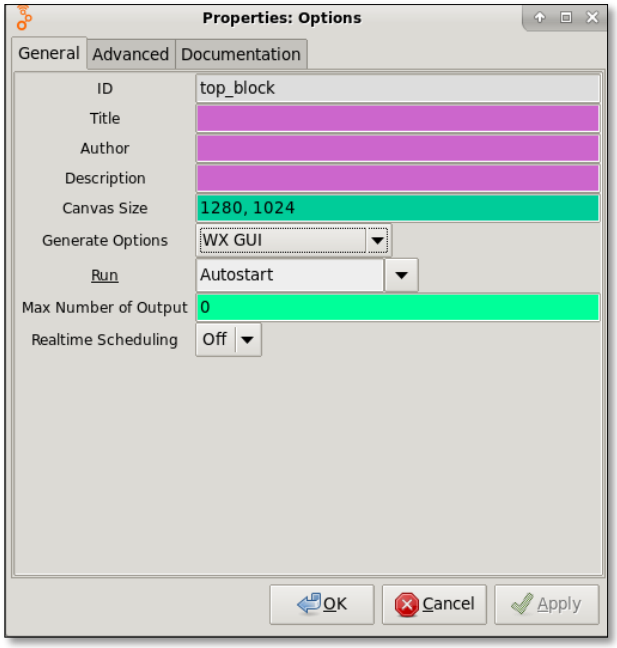


Figura 30 Bloque options – propiedades

Fuente: Investigador

2.- Variable.- Variable permite ingresar el valor de la frecuencia de muestreo en este caso 8 MHz como se muestra en la figura 31.

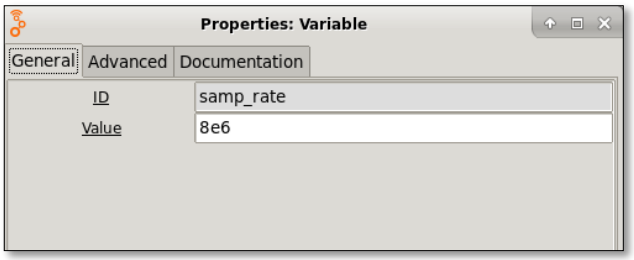


Figura 31 Bloque variable – propiedades

Fuente: Investigador

3.- Osmocom Source.- El Osmocom es un bloque de radio GNU, que le permite utilizar una variedad de hardware SDR con OpenWebRX, que incluye:

- HackRF
- USRP
- RTL-SDR, RTL-TCP
- Dongles DVB-T basados en MSi2500

Esta herramienta se asemeja a una tarjeta programada en la cual se coloca características principales de la señal como se muestra en la figura 32:

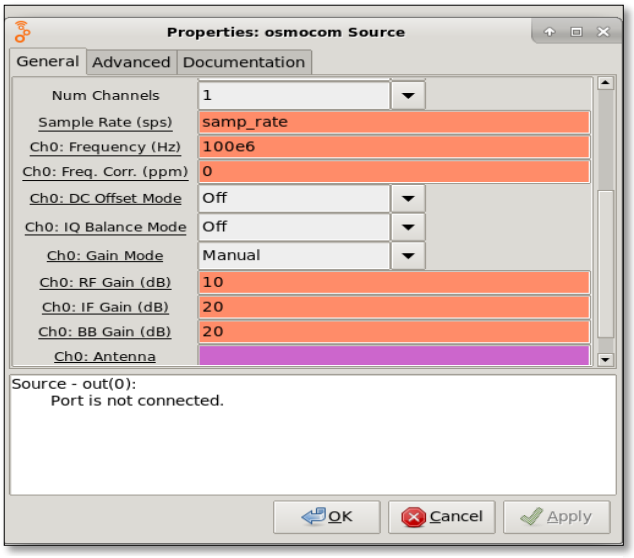


Figura 32 Bloque Osmocom Source – propiedades

Fuente: Investigador

- **Frecuencia.-** Para bluetooth aproximadamente 2.445 GHz
- **Ganancia de radio frecuencia.-** Oscila entre los 10 dB
- **Ganancia de frecuencia intermedia.-** Oscila entre los 20 dB
- **Ganancia de banda de bollinger.-** Es proporcional a la if.

4.- Low pass filter.- El filtro pasa bajo permite atenuar las frecuencias por sobre 2.445 GHz consta de una frecuencia de corte equivalente a 1.9 MHz y ancho de transición de 100 KHz, como se muestra en la figura 33.

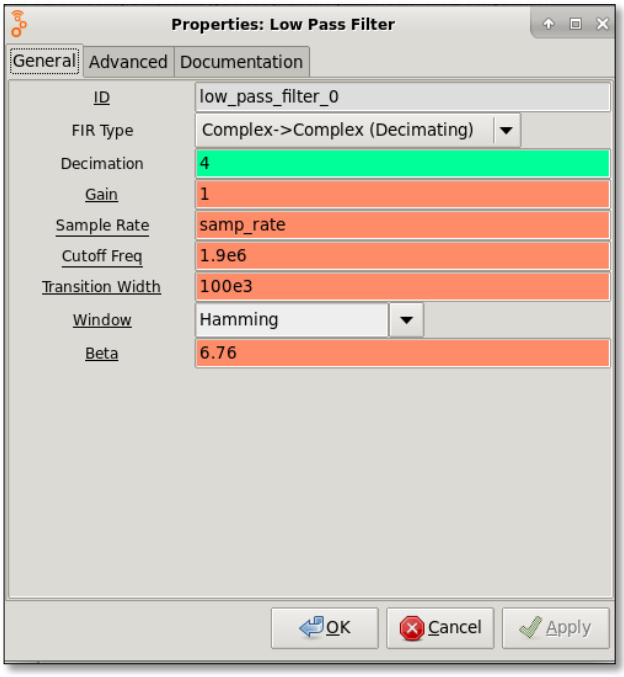


Figura 33 Bloque Low pass filter – propiedades

Fuente: Investigador

5.- Cuadrature Demod.- Puede ser usado para demodular FM, FSK, GMSK, etc. La entrada es banda base compleja, la salida es la frecuencia de la señal en relación con la muestra nominal multiplicada por la ganancia, como se muestra en la figura 34.

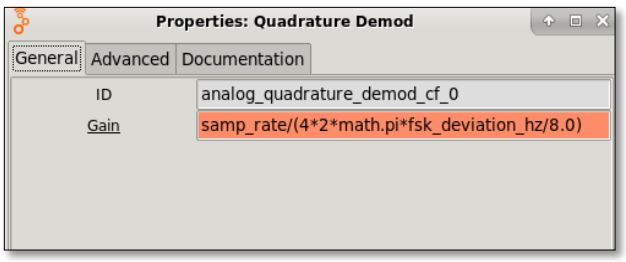


Figura 34 Bloque cuadrature demod – propiedades

.Fuente: Investigador

6. - Float to short.- El float to short convierte un valor en coma flotante (real) en un entero corto de 16 bits, como se muestra en la figura 35, los valores mostrados son predeterminados por el mismo bloque.

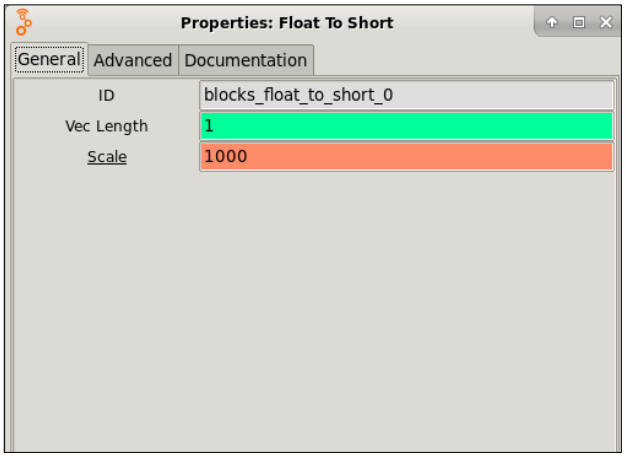


Figura 35 Bloque float to short – propiedades

Fuente: Investigador

7.- Wx GUI scope sink.- La presentación de señal muestra la ventana que contiene la gráfica de la señal captada como muestra en la figura 36. Esta señal es de tipo análoga con alto nivel de interferencia.

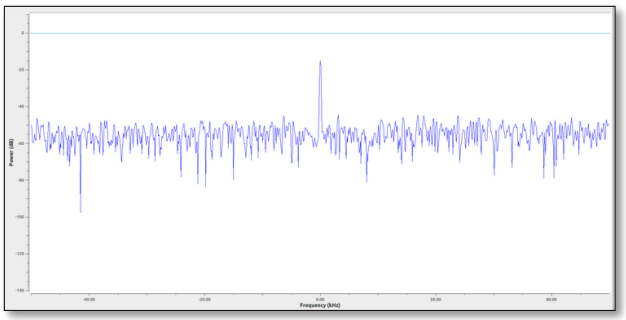


Figura Wx gui scope sink – propiedades

Fuente: Investigador

8.- File Sink.- File sink es un argumento que permite guardar todos los pulsos de señal que se generan al ejecutar una aplicación. Debe darse una ruta de destino para guardar, como se muestra en la figura 37.

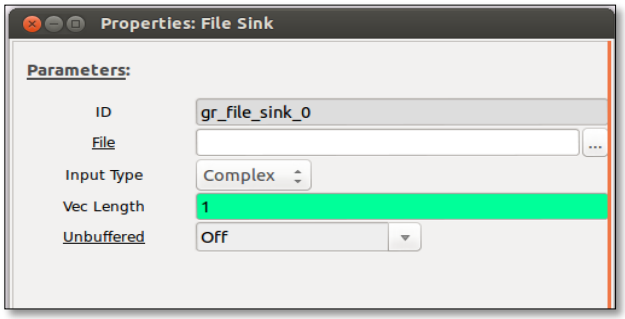


Figura 36 Bloque file sink– propiedades Fuente: Investigador

9.- Variable.- La variable le añade bandas laterales a la frecuencia central 2.4 GHz

10.- Variable.- Desviación a la modulación a un rango de frecuencia de 170 KHz

12.- Variable.- La variable indica el **n**úmero de canales con salto de 1MHz, llegan hasta 45

4.4.2 Flujograma de recepción de señal de control vehicular (Z wave)

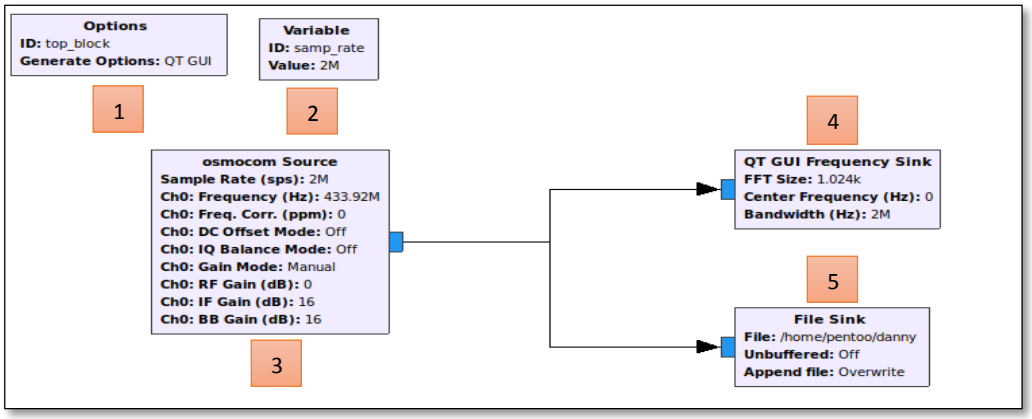


Figura 37 Diagrama de captación de señal Z wave

Fuente: Investigador

La figura 38 muestra un sistema de comunicación mediante SDR para la captación de señal de bloqueo o desbloqueo de puertas de un vehículo. El flujograma indicado tiene dos bloques bien definidos el primero: la fuente que genera la señal, la forma de gráfica de la señal y el destino del archivo donde se guarda la señal captada. El segundo

es un bloque (2) que con tiene la frecuencia de muestreo. A continuación se detalla cada uno de los bloques utilizados.

3.- Osmocom Source.- Osmocom representa el equipo en físico dentro de la simulación, como se muestra en la figura 39.

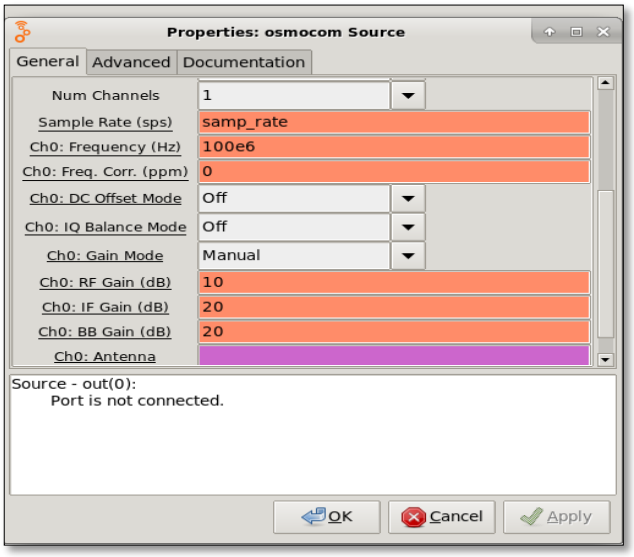


Figura 38 Bloque Osmocom Source – propiedades

Fuente: Investigador

- **Frecuencia.-** Frecuencia para alarma vehicular pavilon 432.75 MHz
- **Ganancia de radio frecuencia.-** La ganancia oscila entre los 10 dB
- **Ganancia de frecuencia intermedia.-** La ganancia de la frecuencia intermedia oscila entre los 20 dB
- **Ganancia de banda de bollinger.-** La ganancia de banda es proporcional a la if.

4.- QT gui Frecuency sink.- El QT contiene varios bloques de interfaz gráfica de usuario basados en QT que agregan receptores gráficos a un diagrama de flujo de Radio GNU. Esta señal referencial es análoga y se mantiene activa esperando algún suceso en su alrededor, como se muestra en la figura 40.

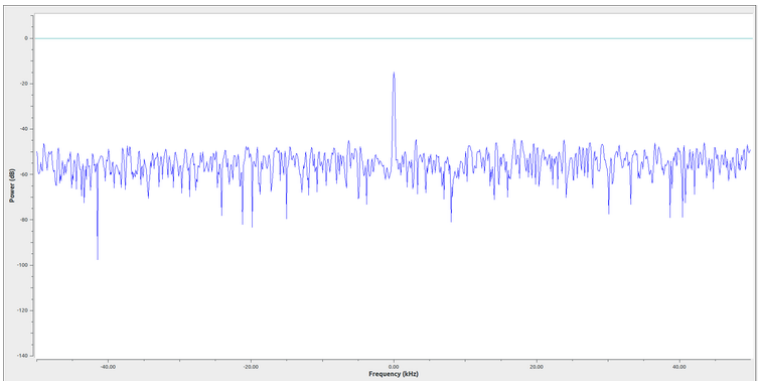


Figura 39 Bloque QT gui Frecuency sink

Fuente: Investigador

5.- File Sink.- File sink es un archivo donde se recepta los valores de frecuencia captados por el dispositivo, envía valores de datos brutos en formato binario al archivo especificado como se muestra en la figura 41. Este archivo sirve para reproducir la señal origen en otro diagrama de flujo llamado transmisión.

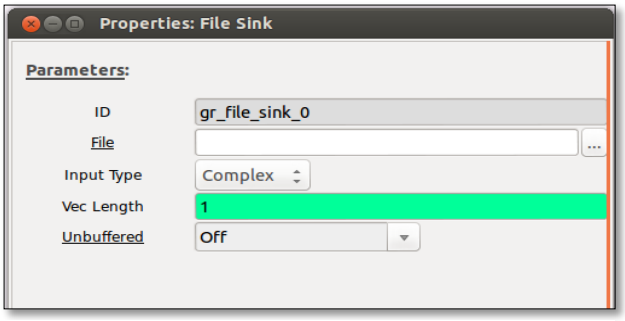


Figura 40 Bloque file sink – propiedades

Fuente: Investigador

Generalmente se guarda el archivo en la misma ubicación del flujograma con el mismo nombre para poder referenciarlo al momento de la réplica en la trasmisión, no tiene ninguna extensión pero si se sobre escribe cuando se ejecuta el programa.

Al compilar se genera automáticamente el código Python asociado al mismo en un archivo *.py, disponiéndose así, de un archivo que permite ver y modificar directamente el código a voluntad del programador.

Al ejecutar el programa se despliega una interfaz de señal de referencia caracterizada por una alta interferencia, sin embargo está lista para poder captar un pulso de señal de acuerdo a la cercanía del mismo, este pulso se reflejara en cualquier posición del pico más alto de señal de referencia (izquierda o derecha) no alcanzara la amplitud máxima, a menos que la señal de referencia coincida con la señal de pulso como se muestra en la figura 42, sin embargo este posicionamiento es importante para poder saber si la frecuencia de operación es la correcta o estamos cerca de encontrarla.

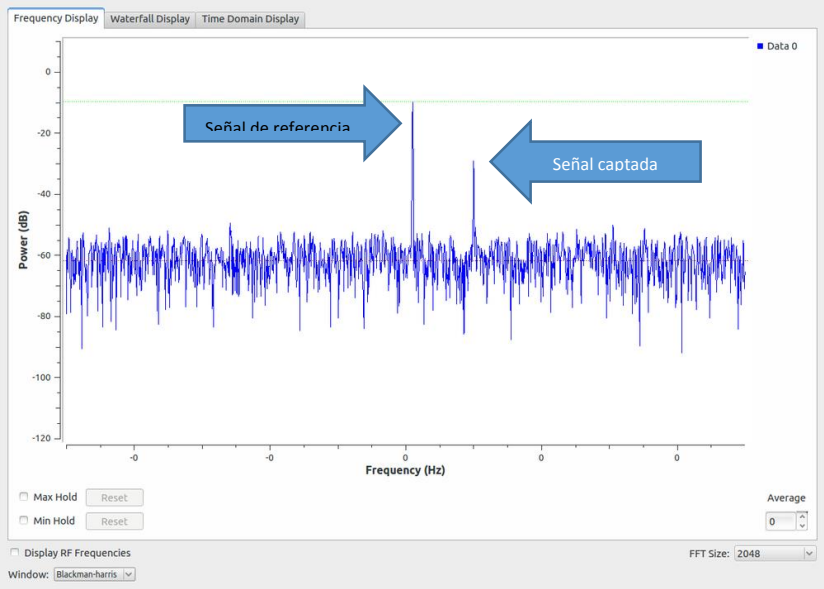


Figura 41 Bloque file sink – propiedades

Fuente: Investigador

Una vez que se recepta la señal de control vehicular la gráfica 42 indica dos señales diferenciadas por su amplitud la una con más amplitud es la señal de referencia, la señal posicionada ligeramente a la derecha es la señal captada. El nivel de amplitud de la segunda señal (captada) es aceptable la prueba se realizó a una distancia aproximada de 2 metros entre el control remoto y el vehículo.

Al realizar el paso anterior la segunda señal queda guardada automáticamente en el archivo binario (file sink) (se debe cerrar la interfaz gráfica). El tiempo que toma el proceso anterior oscila entre los 5 y 7 segundos.

4.4.3 Estructura de flujograma como parte del proceso de hacking Wi-Fi

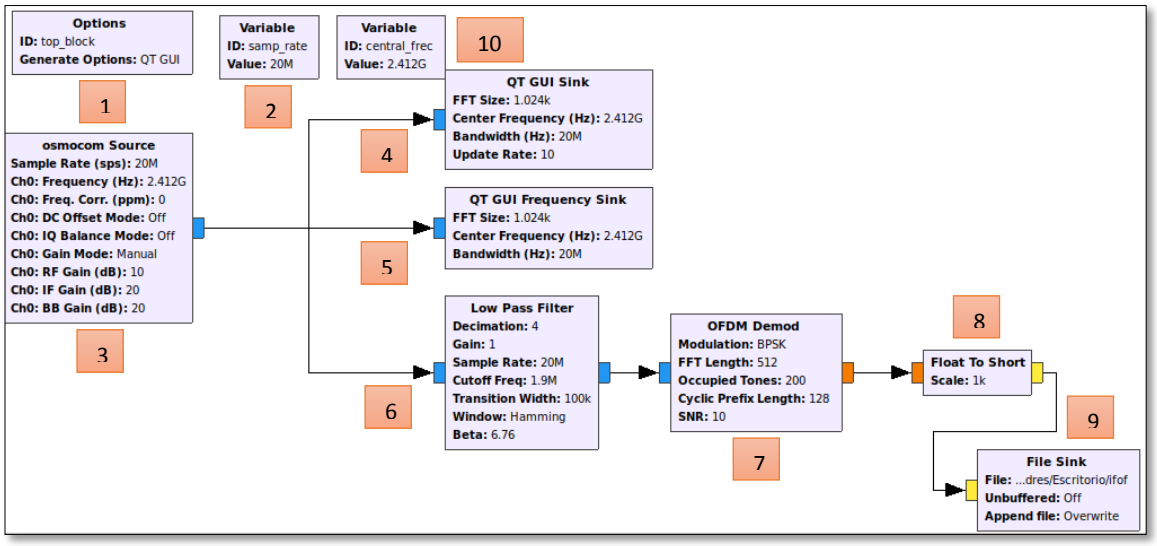


Figura 42 Diagrama de captación de señal Wifi

Fuente: Investigador

La figura 43 indica un sistema de comunicaciones basado en SDR que permite captar una señal wifi cuando se realice una petición cliente –servidor. El diagrama indicado consta de dos bloques estructurados, el primero está compuesto de: generador de señal, filtros, indicadores de señal. El segundo bloque tiene: frecuencia de muestreo y frecuencia central. A continuación se detalla cada uno de los bloques que componen dicho diagrama.

4.- QT gui Sink.- QT gui se considera como la interfaz principal para mostrar señales en diferentes modos como:

- Señal en el dominio frecuencia
- Señal en el dominio tiempo
- Constelación de la señal

Esta interfaz de presentación se muestra como una sola con la característica de verificar el estado de la señal a dependencia del usuario, como se muestra en la figura 44. Las señales indicadas en dicha figura aparecen por defecto cuando se ejecuta el flujograma y son de tipo análoga.

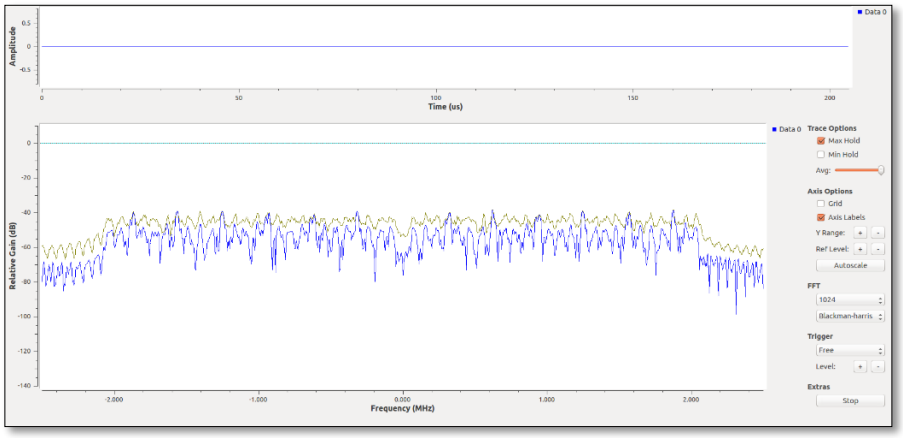


Figura 43 Bloque QT gui Sink

Fuente: Investigador

7.- OFDM Demod.- Demodula una transmisión OFDM recibida en función de las opciones fft_lenght, occupied tones, cp_lenght este bloque realiza la sincronización, FFT y la modulación de los símbolos entrantes OFDM y pasa los paquetes a la capa superior, la entrada es banda base compleja, cuando los paquetes son demulados son enviados a la aplicación mediante callback. En la figura 45 se puede observar algunas características propias de la señal: tipo de modulación, ancho de la transformada rápida de Fourier, el tipo de variable a utilizar.

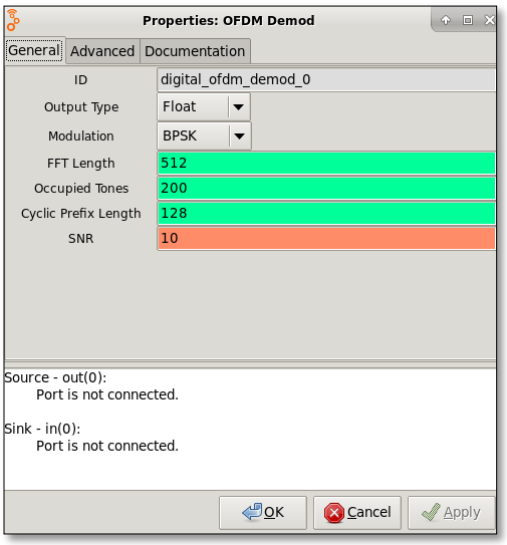


Figura 44 Bloque ofdm demod – propiedades

Fuente: Investigador

8.- Variable.- La variable permite establecer la frecuencia central de la tecnología Wifi 2.412 GHz

FASE DE TRANSFERENCIA

4.5 Diseño de aplicación de hacking ético al IoT para presentación de datos obtenidos.

Para el diseño de la aplicación se presenta una comparativa de diferentes softwares de programación como muestra la tabla 7:

Tabla 8 Comparativa entre leguajes de programación

	Eclipse	Netbeans	Visual Estudio
Generalidades	Dispone de un editor de texto con resaltado de sintaxis. La compilación es en tiempo real. Tiene pruebas unitarias con JUnit, control de versiones con CVS, integración con Ant, asistentes (wizards) para creación de proyectos ,clases, test , etc y refactorización	Edición JavaScript, soporte para usar estructuras Spring de soporte web, integración de MySql más ajustada y una mejor manera de compartir librerías entre proyectos dependientes. El aclamado soporte para RUBY/JRUBY ha sido mejorado con nuevo editor de soluciones rápidas (Quick Fix), un administrador para la plataforma RUBY, soporte para depuración	Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta múltiples lenguajes de programación tales como C++, C#, Visual Basic .NET, F#, Java, Python, Ruby, PHP; al igual que entornos de desarrollo web como ASP.NET MVC,
Tamaño	150 MB (158.286.533 bytes)	79,3 MB (83.155.496 bytes)	219,3 MB (219.155.496 bytes)
Tiempo de descarga	23 minutos	14 minutos	60 minutos
Ejecución	Pc 1.8 Ghz y 6 Gram (13,25 segundos)	Pc 1.8 Ghz y 6 Gram (11,20 segundos)	
Entorno	El entorno de desarrollo integrado (IDE) emplea módulos	Se usa para desarrollar cualquier aplicación	Compatibilidad múltiple
Uso	Es fácil de usa, ligero y estable	Es inestable lento y pesado además de que el código no se puede manipular al 100 porciento	Es fácil de usa, ligero y estable
Disponibilidad	Todos son sistemas multiplataforma para 32 Y 64 bits en Windows , Linux y Mac os		
Complejidad	Más rápido	Más pesado	Más pesado
Flexibilidad	Más flexible	Herramienta Swing estándar	Muy flexible
Herramientas	Más plugins	GUI más intuitivo GIT	GUI más intuitivo GIT
Desarrollo de apps	Mejor soporte de desarrollo Android	SVC más intuitivo Git	Multiplataforma
Pag Web	Herramienta GUI SWT requiere que las bibliotecas nativas se incluya con el productor final	Mejor soporte para PHP	Mejor soporte para PHP

Interoperabilidad		Permite importar proyectos de Eclipse y otros IDE's	Permite importar proyectos de Eclipse y otros IDE's
Add de herramientas	Eclipse no lo trae, pero se le puede añadir como plugins	Netbeans trae por defecto un editor visual con el que podrás crear pantallas de una forma muy sencilla e intuitiva	Herramientas de desarrollo incorporadas

La selección entorno de desarrollo de programación para presentación de datos se basa en la facilidad de programación (estructural), Eclipse es pre configuración de ventanas y editores, relacionada entre sí, y que permiten operar en un determinado entorno de trabajo de forma óptima. Los formularios programados sirven para dar una mejor presentación al proyecto de investigación.

Formulario principal de captación y presentación de datos.

Figura 45 Diseño – formulario principal

Fuente: Investigador

La ventana principal muestra un menú con las tres tecnologías de estudio como se muestra en la figura 46, cada una de ellas cuenta con:

- Interfaz individual de presentación
- Fase 1 del diagrama de hacking (recepción de datos mediante sdr)
- Enlace a un documento .pdf que muestra un reporte del análisis de los datos obtenidos en cada tecnología.

Formulario para Bluetooth

La figura 47 y 48 muestra un diseño para captación y presentación de datos en la tecnología bluetooth y wifi respectivamente. La ventana consta de dos botones donde se puede ejecutar la primera y última fase del proceso de hacking.

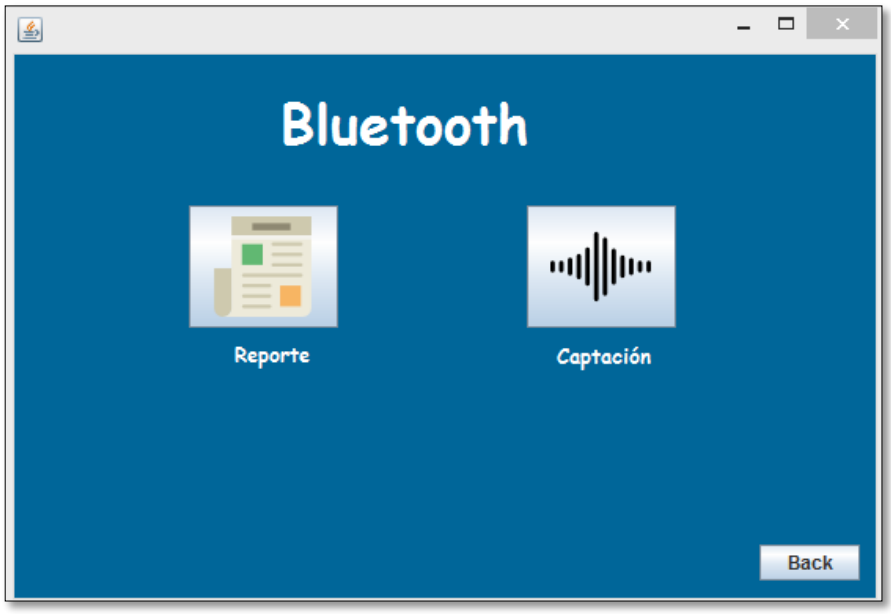


Figura 46 Diseño – formulario bluetooth

Fuente: Investigador

Formulario para Wi_Fi

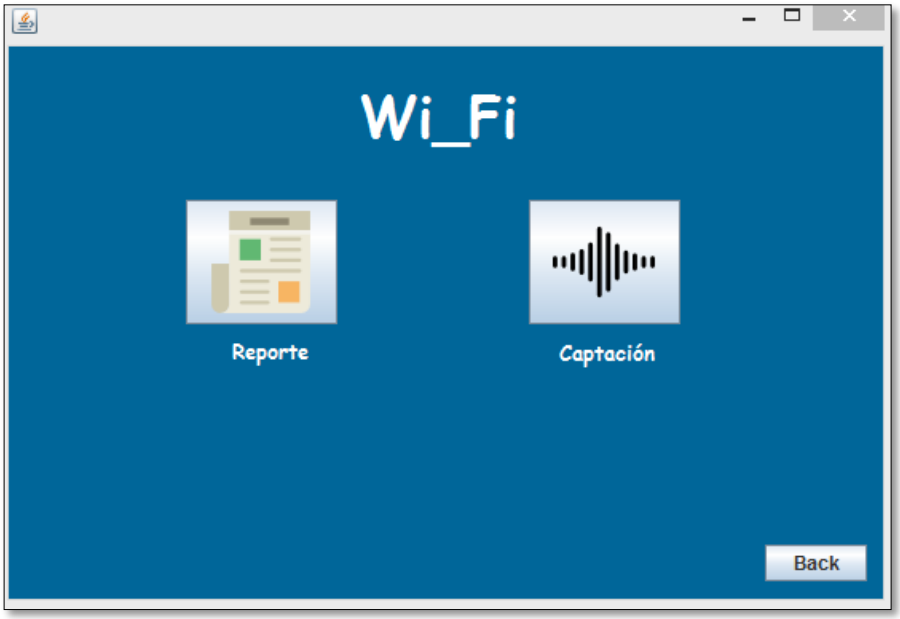


Figura 47 Diseño – formulario Wifi

Fuente: Investigador

Z Wave

A diferencia de las tecnologías anteriores esta interfaz tiene una aplicación adicional; la réplica de la señal vehicular, que está sujeta al proceso de hacking como parte de ejecución como se muestra en la figura 49.

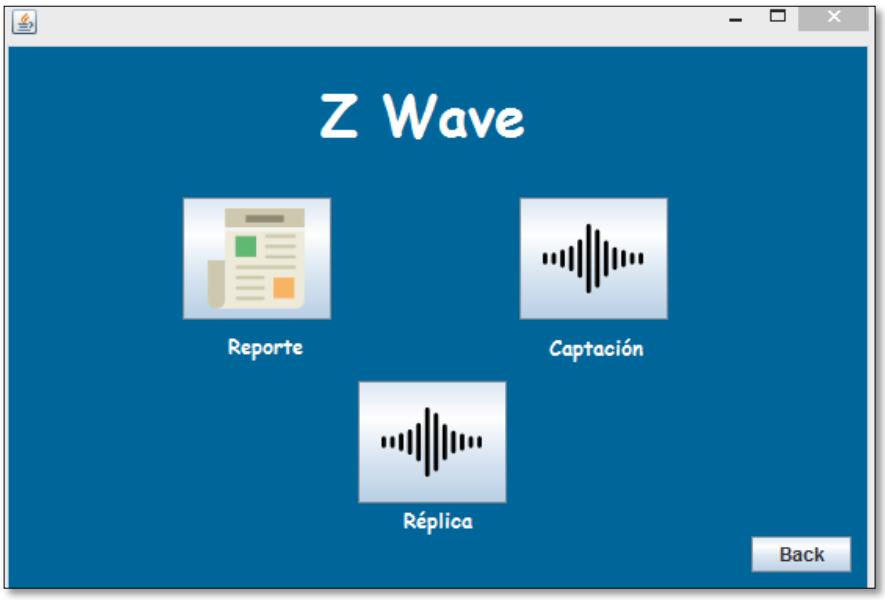


Figura 48 Diseño – formulario Z wave

Fuente: Investigador

4.6 Wireshark como herramienta de análisis en las tecnologías Wifi y Bluetooth

Los datos obtenidos en la fase de acceso son archivos que contienen información de cualquier tipo codificada en binario para el propósito de almacenamiento y procesamiento, estos datos deben ser tratados de manera que se los pueda manipular fácilmente para un posterior análisis, para esto interviene una herramienta denominada Wireshark que es un analizador de protocolos open-Source y que actualmente está disponible para plataformas Windows y Unix, el cual permite mostrar los datos en un lenguaje entendible en el área de la ingeniería en cuestión.

Wireshark implementa una amplia gama de filtros que facilitan la definición de criterios de búsqueda para los protocolos soportados actualmente; y todo ello por medio de una interfaz sencilla e intuitiva que permite desglosar por capas cada uno de los paquetes capturados. Esta herramienta entiende la estructura de los protocolos, se puede visualizar los campos de cada una de las cabeceras y capas que componen los

paquetes monitorizados, proporcionando varias posibilidades a la hora de abordar ciertas tareas en el análisis de tráfico. La interfaz Wireshark se indica en la figura 50.

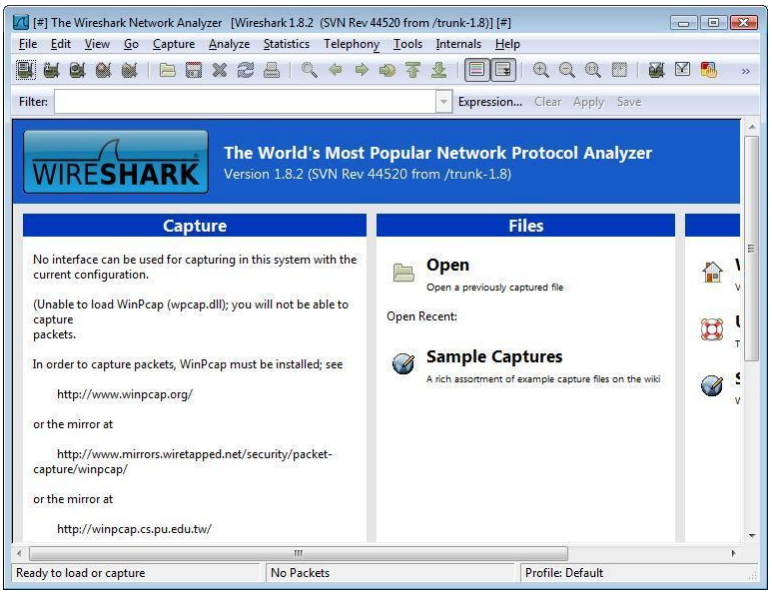


Figura 49 Interfaz principal de Wireshark

Fuente: Investigador

4.7 Diseño de aplicaciones IoT

Para desarrollar el proceso de hacking ético en el proyecto de investigación, fue necesario diseñar e implementar diferentes aplicaciones orientadas al IoT en las tres tecnologías (bluetooth, wifi, Z wave).

4.7.1 Aplicación IoT orientada a Bluetooth

El desarrollo de aplicaciones IoT demanda el uso de hardware y software dependiendo el caso, la figura 51 indica un diseño básico de una aplicación para esta tecnología: control de encendido y apagado de luz mediante bluetooth para la cual se requiere:

- Placa de desarrollo de hardware
- Modulo bluetooth
- Software controlador (IDE Arduino)
- Dispositivo móvil
- Desarrollo de aplicación Android

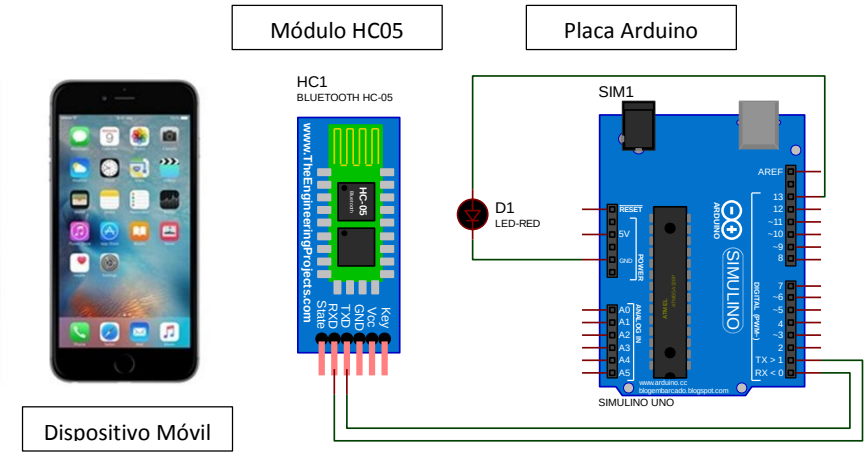


Figura 50 Aplicación motivo de hacking bluetooth

Fuente: Investigador

El diseño de la aplicación Android para el control de luz se realizó en ***MIT app inventor***, este muestra dos interfaces: los bloques de programación y la parte grafica de presentación, este tipo de programación orientada a objetos facilita la parte estructural además de brindar una variedad en diseño en la parte visual como se muestra en la figura 52. Una vez elaborada la aplicación móvil al guardad se construye en modo apk así queda lista para ser instalada en el dispositivo móvil como muestra la figura 53.

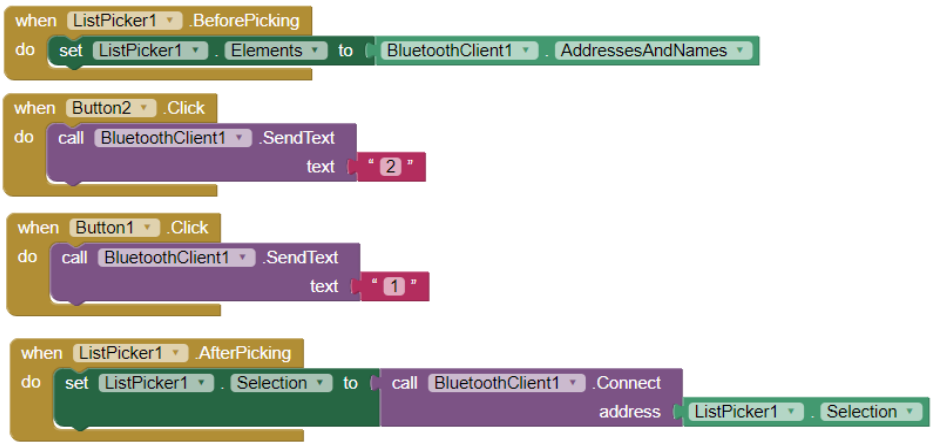


Figura 51 Programación estructural de la aplicación Android

Fuente: Investigador

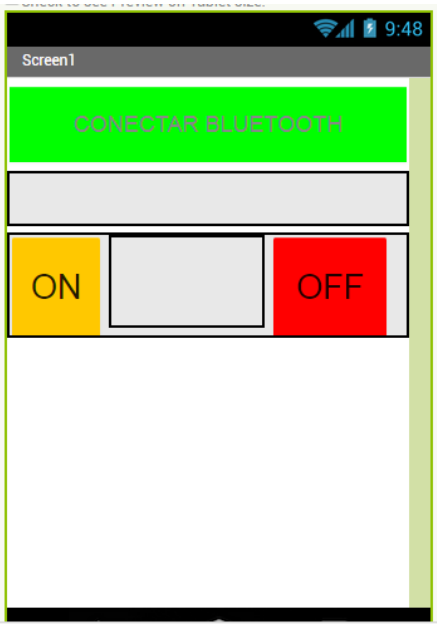


Figura 52 Programación en bloques de la aplicación Android

Fuente: Investigador

Para poder establecer la comunicación inalámbrica mediante bluetooth entre el dispositivo móvil y el módulo HC05 como muestra la figura 5, se programa la placa Arduino con los mismos parámetros de cada acción de los botones de encendido y apagado. No es necesario descargar librerías adicionales para el modulo ya que el dispositivo móvil ejercerá el papel de maestro enviando una señal de apareamiento al módulo en cuestión.

4.7.2 Aplicación IoT orientada a Wi-Fi

La aplicación IoT con el uso de tecnología Wi-fi para el control de encendido y apagado de luz desarrollada requiere:

- Módulo Wi-fi esp8266
- Software controlador (IDE de Arduino)

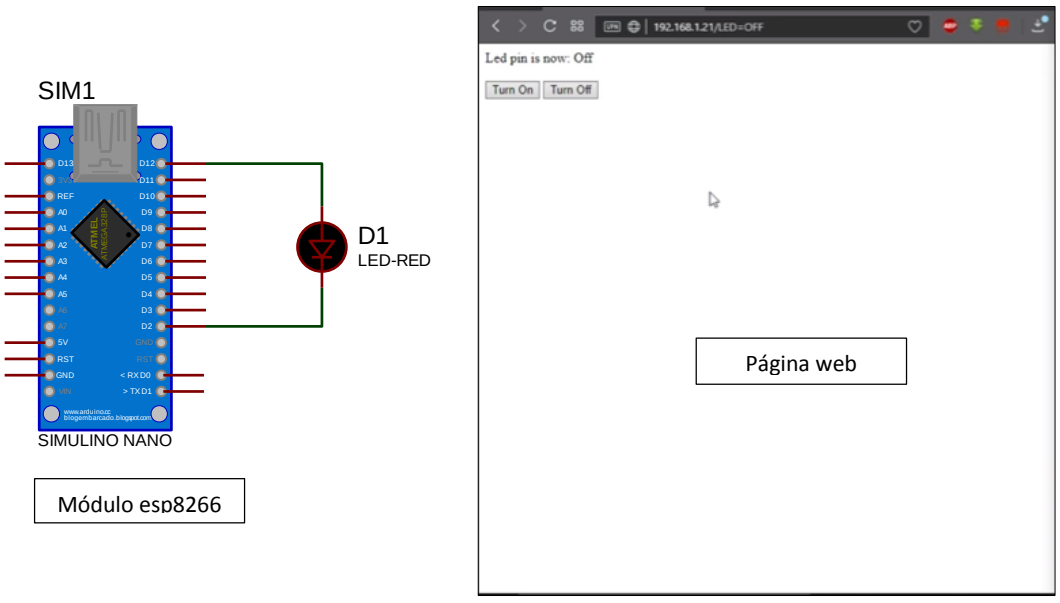


Figura 53. Aplicación motivo de hacking Wifi

Fuente: Investigador

Esta aplicación tiene al igual que la anterior (bluetooth) el control de encendido y apagado de luz vía remota (modo Wi-fi), en la figura 54 se puede observar el diseño de la aplicación para esta tecnología que consta de un módulo y una página web diseñada en HTML, el módulo esp8266 puede ser usado como placa de desarrollo y a la vez como protocolo de comunicación. Para poder programarlo es necesario añadir la librería del módulo al IDE de Arduino debido a que es ahí donde se escribirá el código, se realiza lo siguiente:

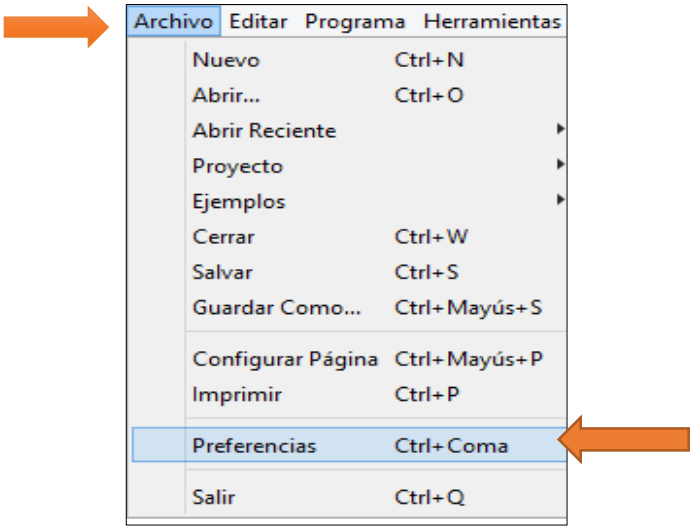


Figura 54 Barra de menú – Ide Arduino

Fuente: Investigador

- Se abre el menú y se busca preferencia en la barra de herramientas, este paso se indica en la figura 55.
- Al ingresar se despliega una ventana donde se escribe: ***http://arduino.esp8266.com/stable/package_esp8266com_index.json*** se guarda cambios antes de salir, como indica la figura 56.

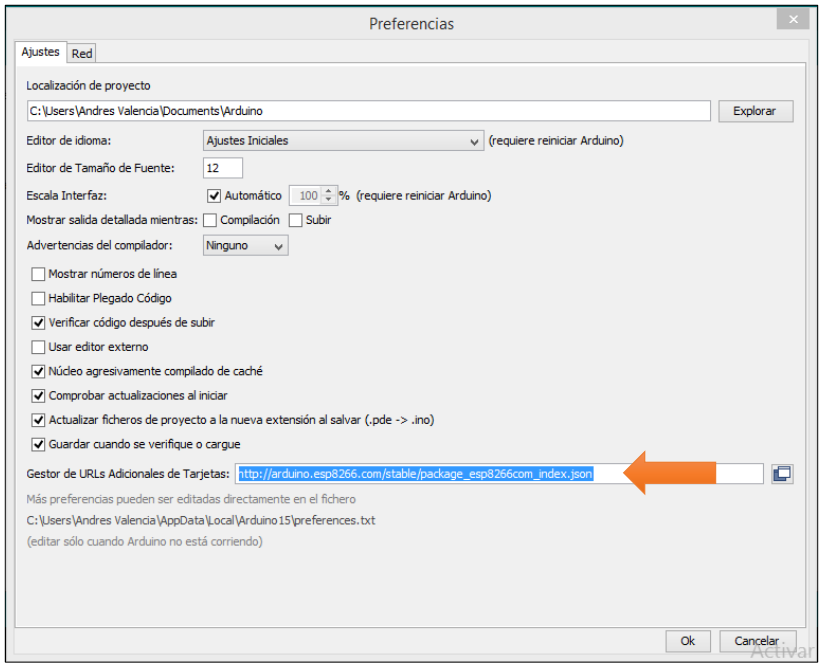


Figura 55 Preferencias – Ide Arduino

Fuente: Investigador

- En la barra de tareas se busca /herramientas/placas/gestor de placas y aparece una ventana como indica la figura 57 donde se instala la librería ya direccionada con el código escrito en el paso anterior.

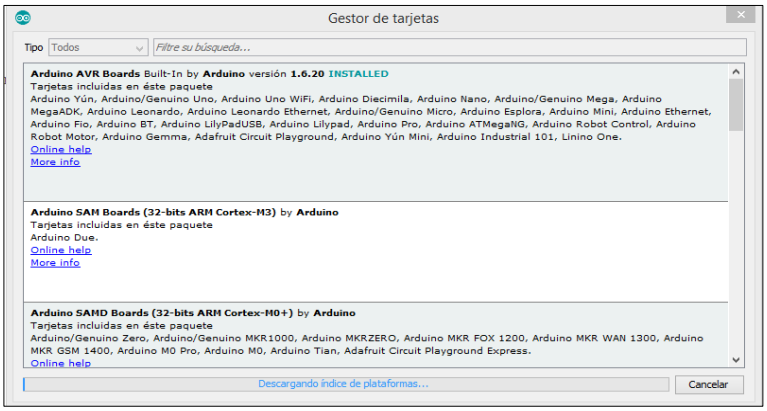


Figura 56 Gestor de tarjetas – Ide Arduino

Fuente: Investigador

- Se instala la última versión y el modulo queda compatible para poder escribir el código que permitirá ejecutar la aplicación.

4.7.3 Replica de señal de alarma de vehículo a 433Mhz (Z wave)

La frecuencia de la señal de un control manual de seguridad vehicular oscila entre los 315 a 433.92 MHz, aunque no es la frecuencia máxima con la que trabaja Z-wave (875 MHz) el principio demótico de alarma es similar en los dos casos, por esta razón el desarrollo de réplica de señal va dirigido a esta tecnología.

Descripción del control remoto a utilizar

El dispositivo inalámbrico positrón diseñado por la empresa con el mismo nombre es un sofisticado sistema electrónico desarrollado con la más avanzada tecnología. Positrón es referente en calidad y tecnología en la provisión de soluciones en telemática, información y confort, además de ser líder en seguridad vehicular, la tabla 8 muestra características técnicas del dispositivo.

Tabla 9 Características del control remoto utilizado

Módulo de alarma	**Mínimo**	**Típico**	**Máximo**
Tensión de operación	9V	12V	16V
Consumo(min=desactivado y Max= Activado)	4mA	-	15mA
Salida de giros	Hasta 3 lámparas de 21 W por salida		
Bloqueo (Px y Fx)	-	-	20A
Salida de señal (Audio,Auxiliar,Traba,etc)	-	-	200mA
Control Remoto			
Frecuencia de operación (MHz)	433.92		

Fuente: Investigador

Para el desarrollo de réplica de señal se necesita dos flujogramas: uno de recepción y otro de envió como indican las figuras 30 y 79 respectivamente, para esto se aplica el diagrama de bloques de fases de hacking, tomando en cuenta la frecuencia exacta del dispositivo.

4.7.4 Análisis Bluetooth

La pila del protocolo Bluetooth incluye protocolos específicos como por ejemplo Link Manager Protocol (LMP) y Logical Link Control and Adaptation Protocol (L2CAP) junto a protocolos no específicos como Objects Exchange Protocol (OBEX) o User Datagram Protocol (UDP). Aparte de todos estos protocolos, la especificación Bluetooth define el Host Controller Interface (HCI), que se encarga de proporcionar una interfaz de comandos al controlador de banda base, al gestor de enlace y proporciona acceso al estado del hardware y a los registros de control,como se muestra en la figura 58.

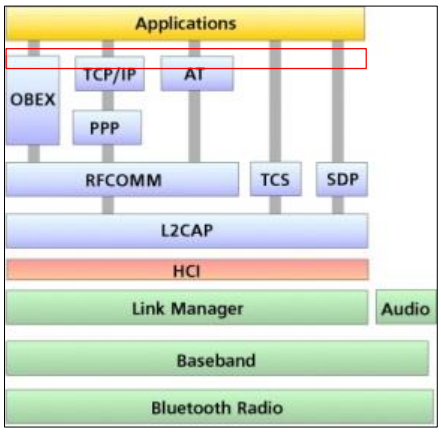
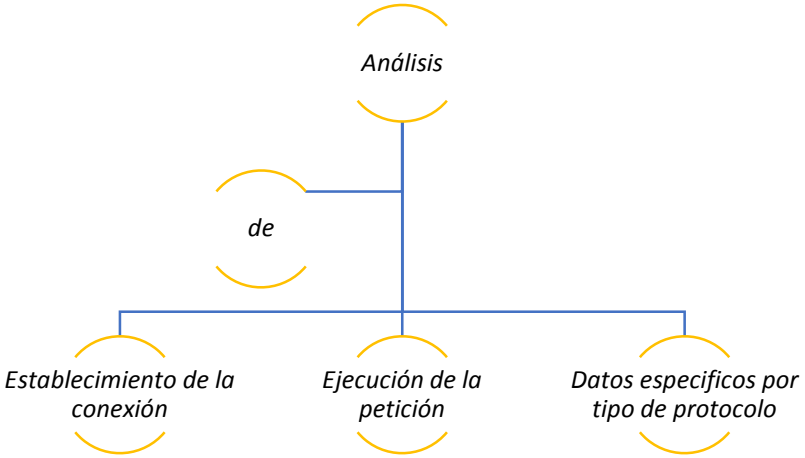


Figura 57 Pila de protocolos bluetooth

Fuente: Investigador

Es precisamente en estos protocolos que recae el análisis puesto que en una comunicación de esta naturaleza (ver aplicación 1). La seguridad es inicializada después de establecerse un canal entre el nivel LM (Link Manager) y el Logical Link Control and Adaptation Protocol (L2CAP).

Para el análisis de datos en la tecnología bluetooth se aplica el esquema presentado a continuación:



Establecimiento de la conexión

El establecimiento de la comunicación entre 2 dispositivos Bluetooth consta de una fase de asociación entre los dos elementos de una red Bluetooth en este caso punto a punto al disponer sólo de dos dispositivos, el esclavo envía continuamente tramas de señalización para notificar de su presencia a un posible maestro y poder establecer una comunicación, la figura 59 indica la trama captada por el analizador (Wireshark).

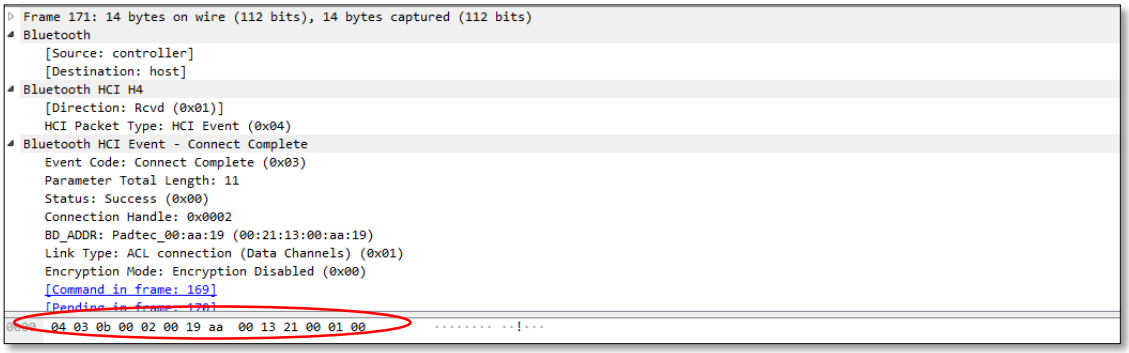


Figura 58 Trama captada por el analizador

Fuente: Investigador

El maestro realiza una fase de descubrimiento con el objetivo de listar todos los dispositivos esclavos a su alcance, identificados gracias a la trama de señalización, para posteriormente enviar un paquete de establecimiento a aquellos con los que quiera comunicarse como se muestra en la figura 60.

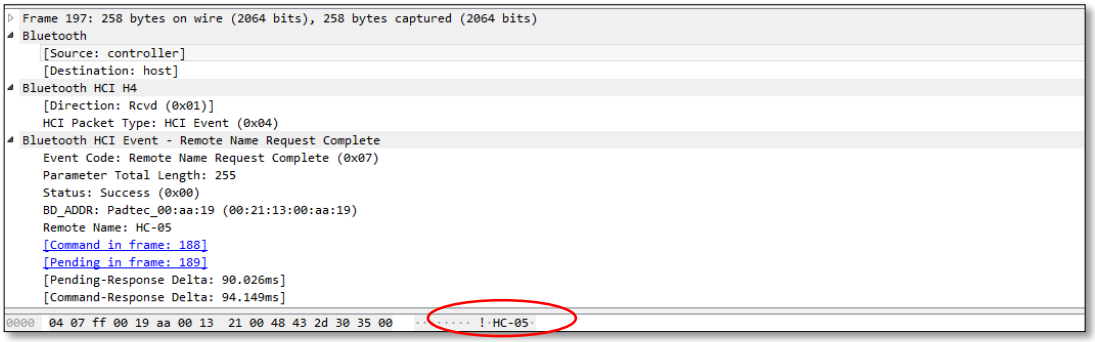


Figura 59 Lista de los dispositivos posibles a conectarse

Fuente: Investigador

La petición de conexión representa el intento de establecimiento por parte del maestro como se muestra en la figura 61, a la que corresponden 2 respuestas:

En la primera respuesta se observa la interacción entre el host y el controlador en el esclavo para tratar la petición realizada por parte del maestro:

- ***Source:*** *Padtec_00:aa:19 (00:21:13:00:aa:19).-* Quien toma la decisión una vez realizada la petición
- ***Destination****: Motorola_ed 42:35 (68:c4:4d:ed:42:35).-* Se envía la respuesta a la petición realizada

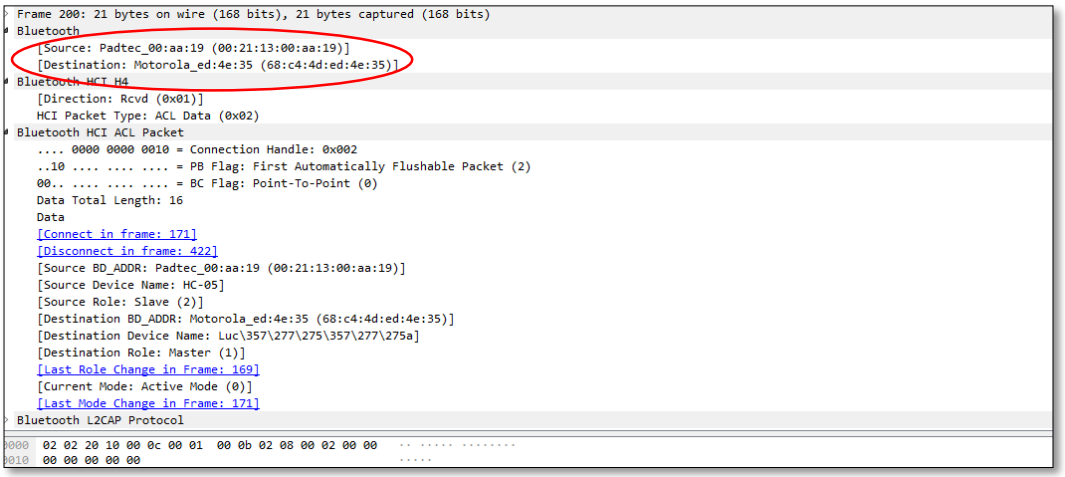


Figura 60 Petición de establecimiento de conexión

Fuente: Investigador

La segunda respuesta se produce finalmente el establecimiento de la conexión, como indica la figura 62.

- ***Source:*** *Motorola_ed 42:35 (68:c4:4d:ed:42:35).-* Confirma la conexión
- ***Destination****: Padtec_00:aa:19 (00:21:13:00:aa:19).-* Recibe la confirmación de conexión

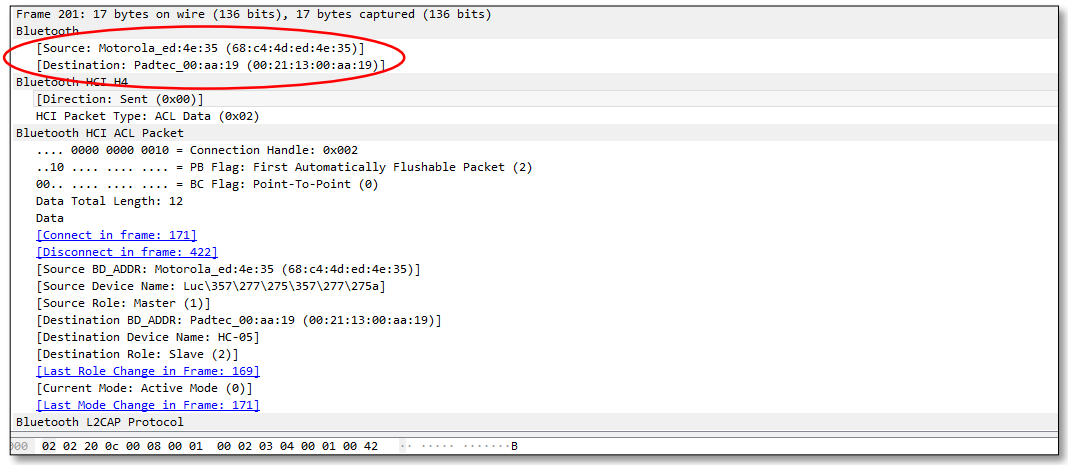
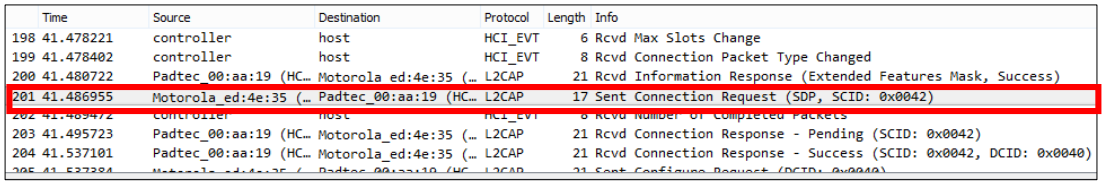


Figura 61 Confirmación de establecimiento de conexión

Fuente: Investigador

Ejecución de la Petición

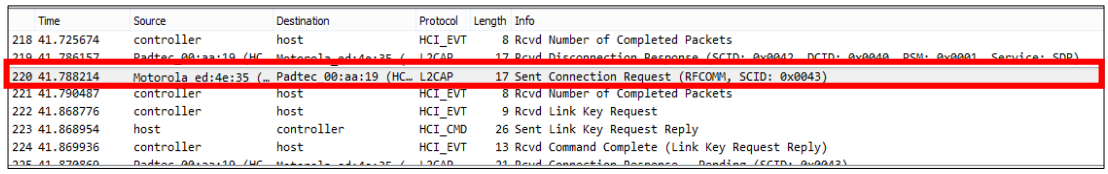
Existen dos capas superiores definidas por L2CAP que son SDP y RFCOM cada una tiene su propio PSM (0x0001, 0x0003) respectivamente, que no es más que la asignación de canales lógicos mediante multiplexación. Una vez establecida la conexión se sincroniza el reloj y se determina el orden de salto en los canales de frecuencia mediante el protocolo L2CAP. Se intercambia información de forma ininterrumpida siguiendo las especificaciones de la tecnología bluetooth con saltos constantes de canal. Cada extremo en un canal se conoce como identificador de canal (CID), en particular estos canales se establecen para las capas superiores a l2cap, la asignación de canales se muestra en la figura 63 y 64.



	Time	Source	Destination	Protocol	Length	Info
198	41.478221	controller	host	HCI_EVT	6	Rcvd Max Slots Change
199	41.478402	controller	host	HCI_EVT	8	Rcvd Connection Packet Type Changed
200	41.480722	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Information Response (Extended Features Mask, Success)
201	41.486955	Motorola_ed:4e:35 (…	Padtec_00:aa:19 (HC…	L2CAP	17	Sent Connection Request (SDP, SCID: 0x0042)
203	41.495723	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Connection Response - Pending (SCID: 0x0042)
204	41.537101	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Connection Response - Success (SCID: 0x0042, DCID: 0x0040)

Figura 62 Identificador de canal destino

Fuente: Investigador



	Time	Source	Destination	Protocol	Length	Info
218	41.725674	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
220	41.788214	Motorola_ed:4e:35 (…	Padtec_00:aa:19 (HC…	L2CAP	17	Sent Connection Request (RFCOMM, SCID: 0x0043)
221	41.790487	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
222	41.868776	controller	host	HCI_EVT	9	Rcvd Link Key Request
223	41.868954	host	controller	HCI_CMD	26	Sent Link Key Request Reply
224	41.869936	controller	host	HCI_EVT	13	Rcvd Command Complete (Link Key Request Reply)

Figura 63 Identificador de canal y salto en una posición

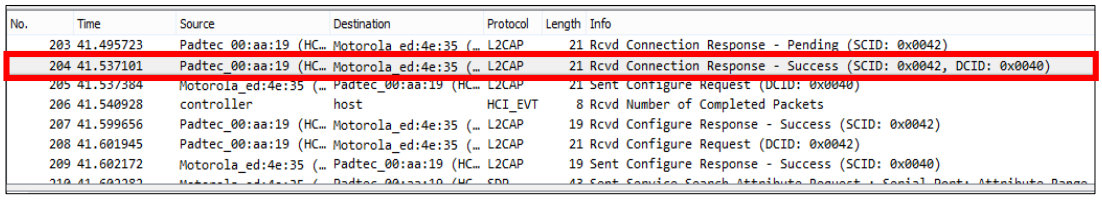
Fuente: Investigador

- Si el paquete L2CAP se origina en un dispositivo local, este extremo de la tubería se llama identificador de canal fuente (SCID).

- Dentro de este paquete se puede observar la asignación dinámica de canal que entra en funcionamiento cuando el maestro envíe un dato, el salto de canal será de una posición

- La asignación dinámica de canal entre maestro – esclavo es: 0x0042 y 0x0043 que tiene su equivalente en decimal a 66 y 67 respectivamente.

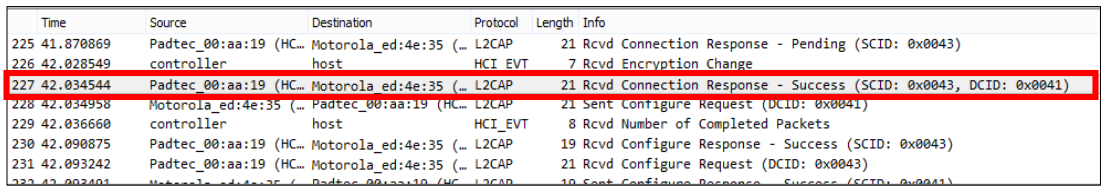
De lo contrario, si el paquete L2CAP es enviado a través del canal a un dispositivo remoto, este extremo del canal se le denomina identificador de canal destino (DCID). Cada vez que el esclavo recibe información por parte del maestro este envía un acuse de recibo para indicar que los datos llegaron satisfactoriamente, es ahí cuando se puede visualizar establecimiento de la conexión, como indican las figuras 65 y 66 respectivamente.



No.	Time	Source	Destination	Protocol	Length	Info
203	41.495723	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Connection Response - Pending (SCID: 0x0042)
204	41.537101	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Connection Response - Success (SCID: 0x0042, DCID: 0x0040)
205	41.537384	Motorola_ed:4e:35 (…	Padtec_00:aa:19 (HC…	L2CAP	21	Sent Configure Request (DCID: 0x0040)
206	41.540928	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
207	41.599656	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	19	Rcvd Configure Response - Success (SCID: 0x0042)
208	41.601945	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Configure Request (DCID: 0x0042)
209	41.602172	Motorola_ed:4e:35 (…	Padtec_00:aa:19 (HC…	L2CAP	19	Sent Configure Response - Success (SCID: 0x0040)

Figura 64 Acuse de recibo por parte del esclavo

Fuente: Investigador



	Time	Source	Destination	Protocol	Length	Info
225	41.870869	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Connection Response - Pending (SCID: 0x0043)
226	42.028549	controller	host	HCI_EVT	7	Rcvd Encryption Change
227	42.034544	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Connection Response - Success (SCID: 0x0043, DCID: 0x0041)
228	42.034958	Motorola_ed:4e:35 (…	Padtec_00:aa:19 (HC…	L2CAP	21	Sent Configure Request (DCID: 0x0041)
229	42.036660	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
230	42.090875	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	19	Rcvd Configure Response - Success (SCID: 0x0043)
231	42.093242	Padtec_00:aa:19 (HC…	Motorola_ed:4e:35 (…	L2CAP	21	Rcvd Configure Request (DCID: 0x0043)

Figura 65 Confirmación de establecimiento de conexión

Fuente: Investigador

En la trama 204 y 227 podemos observar que el esclavo recibe una petición para establecer una canal de comunicación al envió de datos de la siguiente manera:

- El maestro envía la petición por el canal 0x0042 y 0x0043 con la nominación SCID por el efecto salto de canal llega por el 0x0040 y 0x0041 respectivamente

Datos específicos por tipo de protocolo

En el desarrollo de la aplicación bluetooth, al establecer los parámetros de prendido y apagado de luz se toma en consideración para las dos acciones 1 y 2 respectivamente. Estos números en lenguaje C son notados como símbolos que tienen su equivalencia en los diferentes sistemas numéricos de escritura, la figura 67 muestra el equivalente de estos símbolos que serán objeto de análisis:

ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo
0	0	NUL	16	10	DLE	32	20	(space)	48	30	0
1	1	SOH	17	11	DC1	33	21	!	49	31	1
2	2	STX	18	12	DC2	34	22	"	50	32	2
3	3	ETX	19	13	DC3	35	23	#	51	33	3
4	4	EOT	20	14	DC4	36	24	$	52	34	4
5	5	ENQ	21	15	NAK	37	25	%	53	35	5
6	6	ACK	22	16	SYN	38	26	&	54	36	6
7	7	BEL	23	17	ETB	39	27	'	55	37	7
8	8	BS	24	18	CAN	40	28	(	56	38	8
9	9	TAB	25	19	EM	41	29	)	57	39	9
10	A	LF	26	1A	SUB	42	2A	*	58	3A	:
11	B	VT	27	1B	ESC	43	2B	+	59	3B	;
12	C	FF	28	1C	FS	44	2C	,	60	3C	<
13	D	CR	29	1D	GS	45	2D	-	61	3D	=
14	E	SO	30	1E	RS	46	2E	.	62	3E	>
15	F	SI	31	1F	US	47	2F	/	63	3F	?

Figura 66 Tabla de conversión símbolo - sistema

Fuente: Investigador

El archivo que contiene los datos capturados en la fase de acceso lleva consigo los símbolos mencionados anteriormente, al ser enviado a Wireshark esta muestra el símbolo como tal, por otro lado la trama de información presentada al administrador contiene el valor equivalente en hexadecimal la figura 68 muestra en detalle lo descrito:

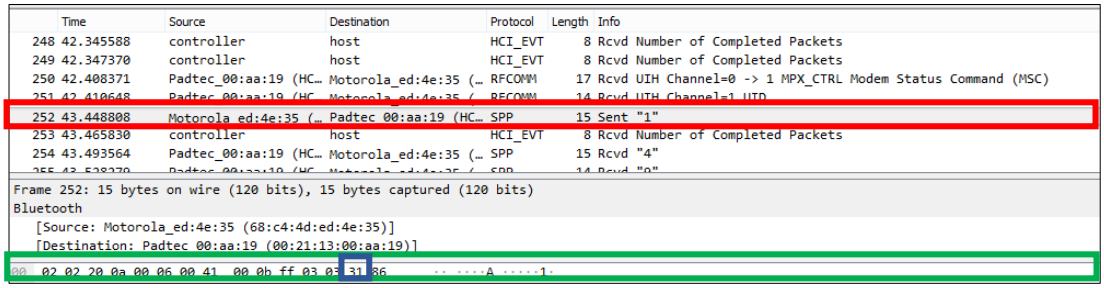


Figura 67 Envió de dato para el encendido de luz

Fuente: Investigador

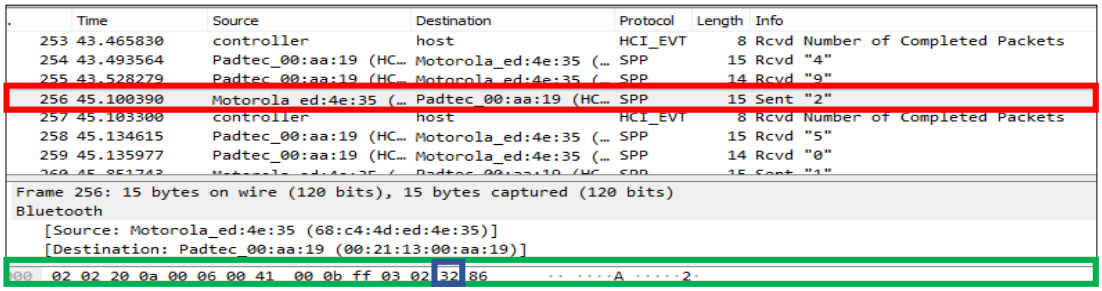


Figura 68 Envió de dato para el apagado de luz

Fuente: Investigador

El análisis de los datos se basa en el tipo de seguridad que posee el modo de operación de la aplicación, se sabe que la seguridad es inicializada después de establecerse un canal entre el nivel LM (Link Manager) y el de L2CAP. Las políticas de seguridad y los niveles de confianza se aplican de forma independiente, permitiendo accesos de aplicaciones con diferentes requerimientos. La figura 70 muestra los protocolos que serán caso de análisis:

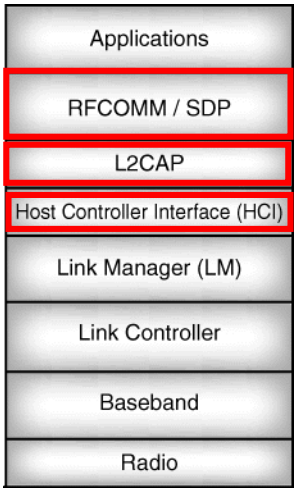


Figura 69 Pila de protocolos involucrados en el análisis

Fuente: Investigador

Como se ve en la figura antes mencionada los protocolos Host Controller Interface (HCI), L2CAP, RFCOM y SPP serán objeto de análisis en la herramienta, la figura 71 muestra el emparejamiento descrito anteriormente:

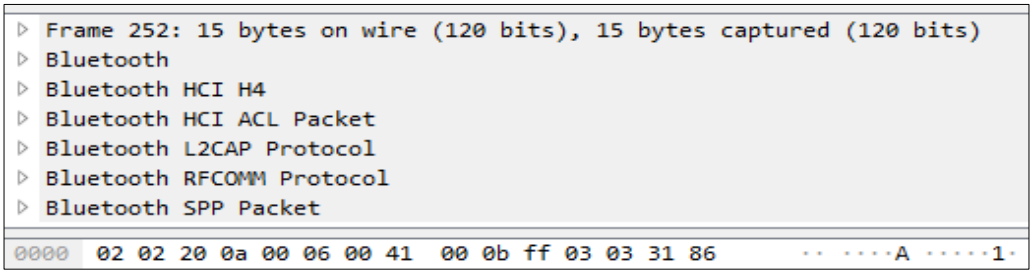


Figura 70 Protocolos captados por Wireshark

Fuente: Investigador

Wireshark muestra información general de toda la trama obtenida, como indica la figura 72:

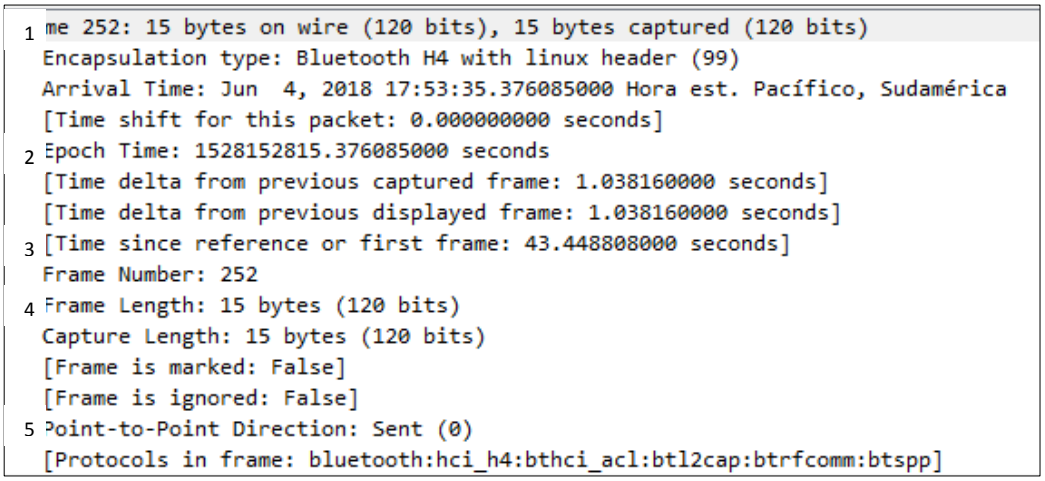


Figura 71 Información de la trama obtenida por el analizador

Fuente: Investigador

1.- Muestra el tipo de encapsulación Bluetooth. Está diseñado para operar a través de un enlace RS232 sin paridad, se requiere control de flujo de hardware, los comandos de HCI se transmiten directamente con la adición de un indicador de paquete para indicar:

- Distintos de conexión sincronizada (SCO)
- Los datos sin conexión asincrónicos (ACL)

2.- Muestra el tiempo de referencia desde la primera trama (inicio de conexión)

3.- Indica el número de trama capturada

4.- Indica la longitud de trama capturada

5.- Muestra los protocolos capturados Hci_l2cap_rfcom_Spp_acl

Bluetooth

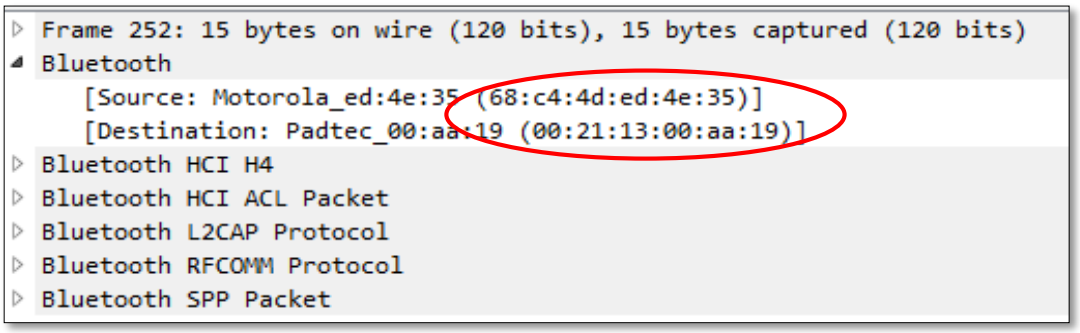


Figura 72 Direcciones MAC del maestro – esclavo

Fuente: Investigador

- Source: Motorola_ed:4e:35 (68:4c:4d: ed: 35).- Indica el nombre y la dirección Mac del dispositivo origen conocido como maestro.
- Destination: Padtec_00:aa:19 (00:21:13:00:aa:19).- Indica el nombre y la dirección Mac del dispositivo destino que cumple la petición del maestro
- Estas nominaciones concuerdan con los elementos usados en el desarrollo de la aplicación 1. Para la descripción se tomó como referencia la figura 73

Bluetooth HCI H4

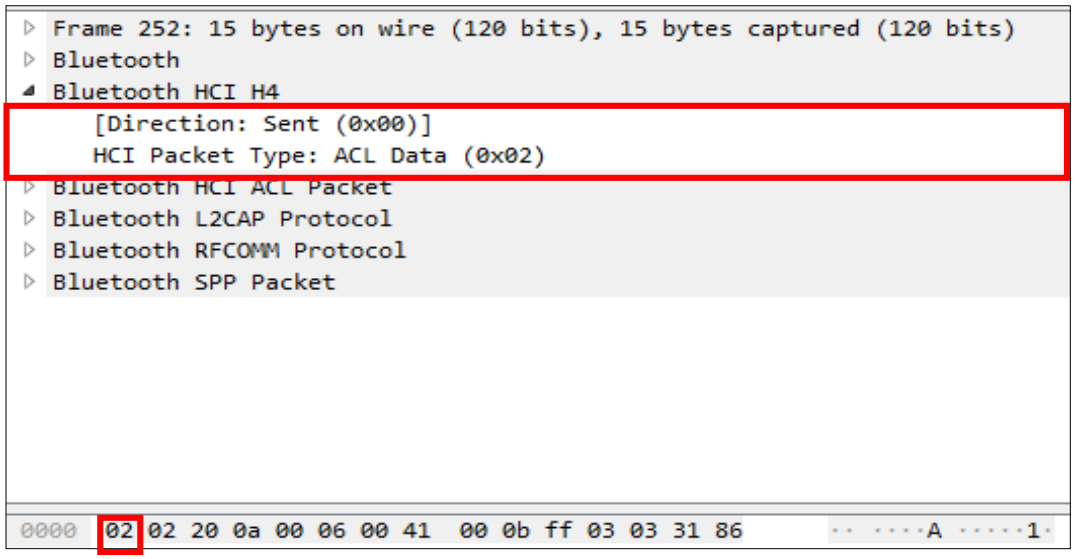


Figura 73 Indica el tipo de paquete HCI

Fuente: Investigador

El paquete HCI se usa para enviar comandos al controlador desde el host. El formato del paquete de comandos HCI se muestra a en la figura 75:

HCI packet type	HCI packet indicator
HCI Command Packet	0x01
HCI ACL Data Packet	0x02
HCI Synchronous Data Packet	0x03
HCI Event Packet	0x04

Figura 74 Tipos de paquete HCI

Fuente: Investigador

- En la figura 74 Wireshark muestra el tipo de paquete HCI con un encabezado propio de ACL donde los paquetes de datos pueden enviarse desde y hacia el controlador de host Bluetooth.
- Trabaja con el tipo de encapsulamiento de los paquetes obtenidos, se puede ver que muestra el tipo de dato enviado y el modo de operación:
    - ACL Data (0x02).- Tipo de conexión al envió de datos y la posición dentro de la trama respectivamente

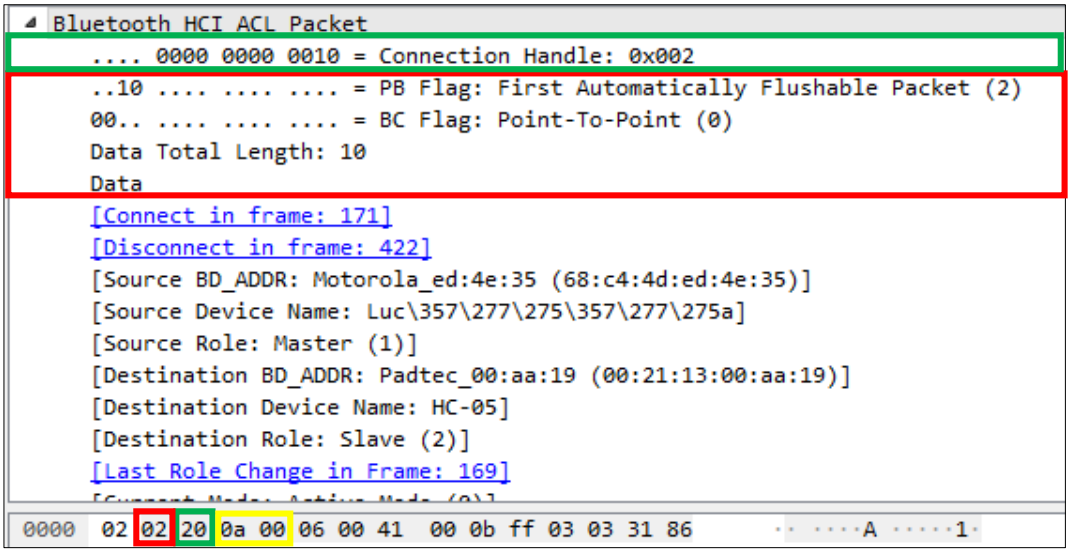


Figura 75 Análisis de datos HCI

Fuente: Investigador

- Connection handle: 0x002.- Se utiliza para identificar una conexión, los bits de referencia a este proceso se colocan en una posición menos significativa tiene un valor que va desde 0x0000 hasta 0xEF00.
- PB Flag ..10 …. …. …. .- Identifica si los datos del paquete llevan el inicio de un paquete L2CAP de capa superior o si es un paquete continuo de un paquete L2CAP.
- BC 00.. …. …. …. .- Identifica los datos punto a punto de difusión y discrimina entre punto a punto, transmisión activa (para esclavos activos) y difusión de piconet (para esclavos activos y empaquetados), el valor de 00 al inicio no significa que no haya esclavos activos, al contrario se puede ver mediante la dirección MAC un esclavo activo. PB flag y BC flag forma el número 0x0010 que en en el fragmento de trama está ubicado en la posición más significativa.
- Se muestra la longitud del resto de la trama notada por posiciones después de la designación (10 ¬ 0a00). La descripción se tomó de la figura 66.

Bluetooth L2CAP Protocol

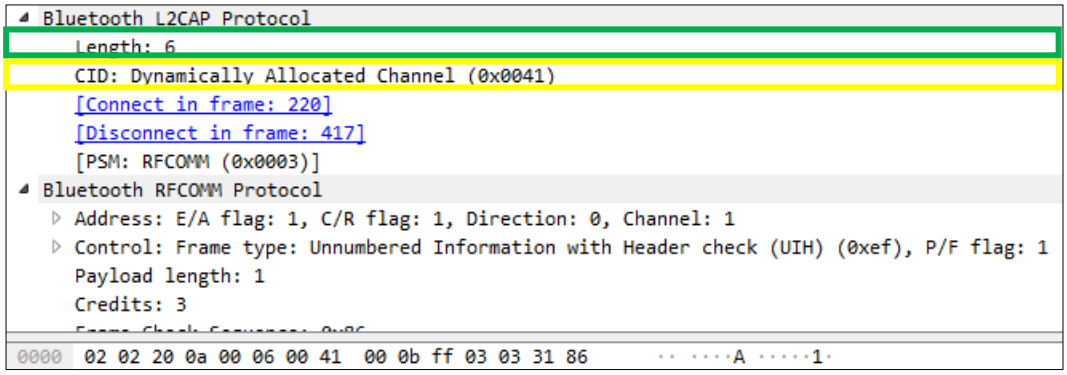


Figura 76 Análisis de datos L2CAP

Fuente: Investigador

- Muestra la longitud del fragmento de trama del protocolo L2CAP que es de 6 notado como 0x0600).
- También indica el canal asignado dinámicamente (0x0041). La descripción se tomó de la figura 77.

Bluetooth RFCOM Protocol

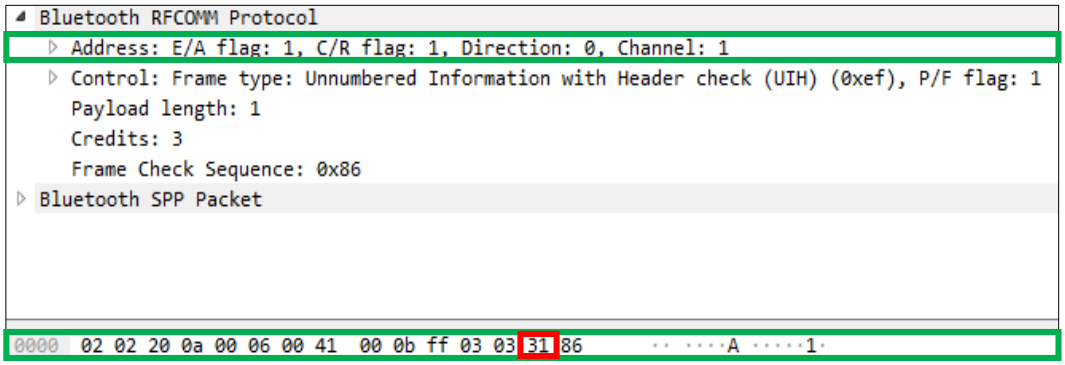


Figura 77 Análisis de datos RFCOM

Fuente: Investigador

- ***Address: E/A flag:1 C/R flag:1.-*** El envío de banderas es un indicador que la comunicación se ha establecido, un paso anterior donde se comprueba este posicionamiento es la visualización de las direcciones MAC de Source y Destination
- ***Direction :0.-*** Como se describe anteriormente la cantidad cero no indica un valor numérico sino parte de una combinación numérica en formato hexadecimal que viene acompañado del canal por el cual llegan los datos a su destino (***chanel:1***) que es el canal 0x0043
- ***Payload: lenght: 1.-*** Muestra la longitud del dato captado (0x0001). La descripción se tomó de la figura 78.

4.7.5 Análisis Z wave

El análisis de esta tecnología se realiza mediante un flujograma indicado en la figura 79 que permite tomar los datos asignados en la fase de recepción:

- Frecuencia de operación
- Frecuencia de muestreo
- Ganancia de frecuencia
- Frecuencia intermedia

Datos que se obtienen mediante la investigación de la señal del dispositivo en cuestión.

Flujograma de transmisión (réplica) de señal de control vehicular

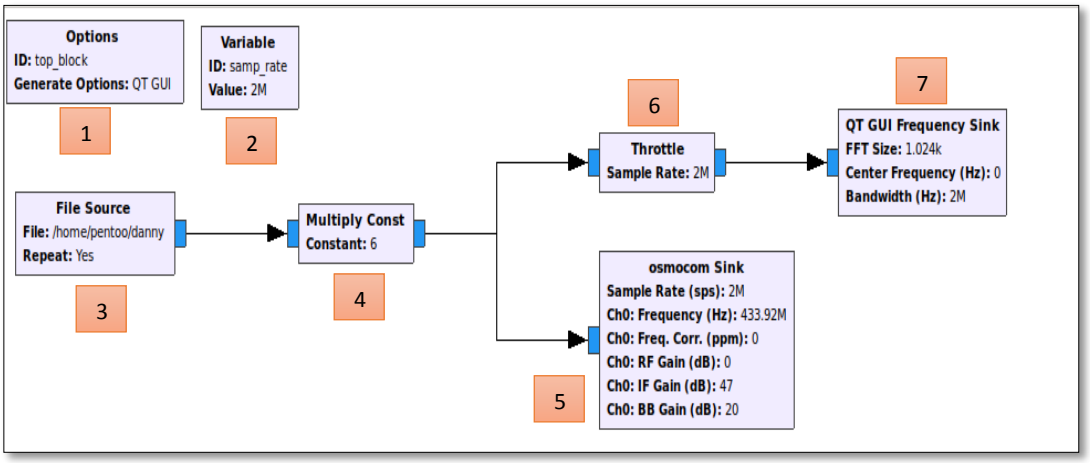


Figura 78 Diagrama de flujo – réplica de señal

Fuente: Investigador

Para esta fase se desarrolla un segundo diagrama de flujo que se diferencia del primero en dos aspectos:

- Tratamiento a la señal para amplificarla.- Uso de bloques adicionales para amplificar la señal y proteger el equipo que contiene el diagrama en ejecución por trabajar con señales en tiempo real.
- Intermitencia del pulso de señal captada en periodos de tiempo cortos.- La señal que se capta en el primer flujograma y que también es parte de la fase 1 de recepción es acompañada de otra señal de menor amplitud que aparece en periodos de tiempo corto que cumple el papel del dispositivo activador o desactivador de cualquier aplicación domótica.

Se presenta el análisis de cada uno de los bloques que intervienen en la réplica de señal, que son:

3.- File Source.- File Source Es el bloquee principal donde se carga el archivo que contiene los pulsos de desbloqueo de control vehicular, como indica la figura 80

Figura 79 Bloque file Source – propiedades

Fuente: Investigador

4.- Multiply Const.- Multiply Const es un bloque amplificador que permite potenciar los pulsos de señal captados a 6 niveles, como indica la figura 81

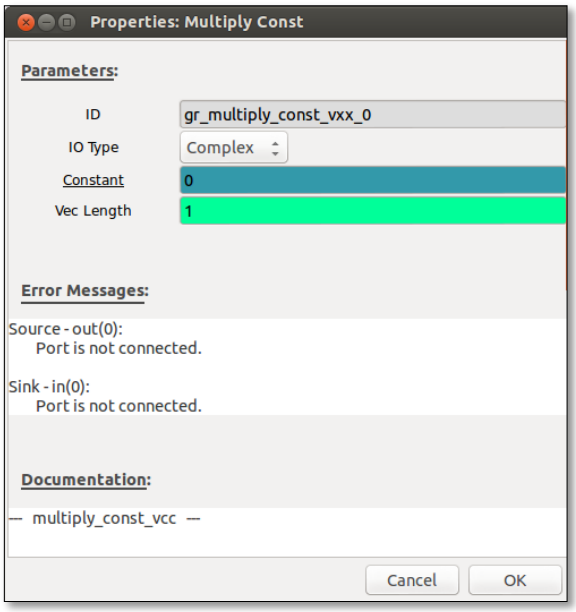


Figura 80 Bloque Multiply const– propiedades

Fuente: Investigador

5.- Osmocom Sink.- Este bloque representa al equipo físico dentro de la simulación, debe contar con las mismas propiedades que ***osmocom,*** como indica la figura 82.

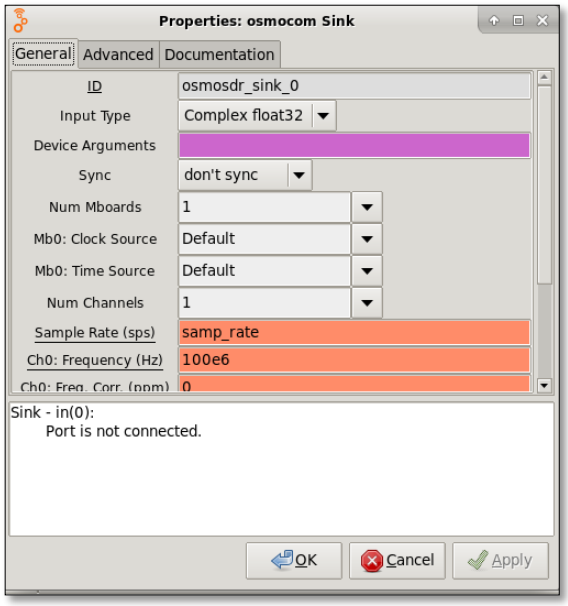


Figura 81 Bloque Osmocom sink – propiedades

Fuente: Investigador

- Esta es la parte donde se escriben por segunda vez los valores característicos de la señal original:
    - Frecuencia de operación: 412MHz
    - Ganancia de operación: 45dBm
    - Ganancia de If: 20dBm
    - Frecuencia de muestreo: 2MHz

6.- Throtle.- El Throtle limita el rendimiento de datos a la velocidad de muestreo especificada. Esto evita que GNU Radio consuma todos los recursos de la CPU cuando el diagrama de flujo no está siendo regulado por hardware externo, como lo indica la figura 83.

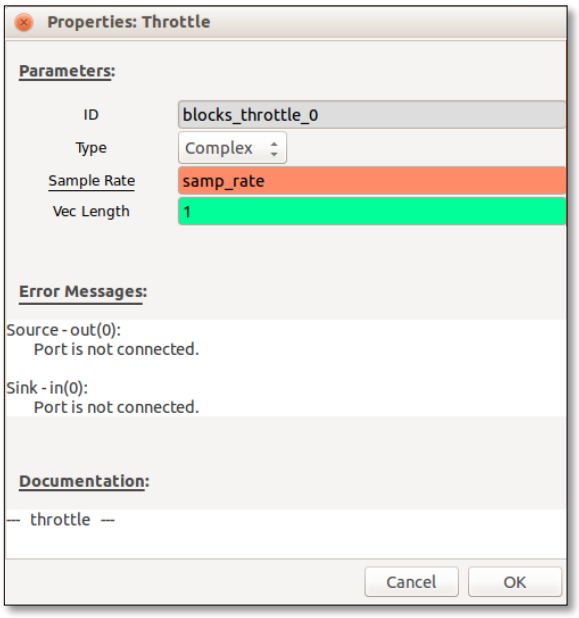


Figura 82 Bloque Throtle – propiedades

Fuente: Investigador

7.- QT GUI Sink.- Muestra la señal análoga duplicada, y claramente se puede observar la señal de referencia y la réplica que actuara inmediatamente se la ejecute. Así lo muestra la figura 84.

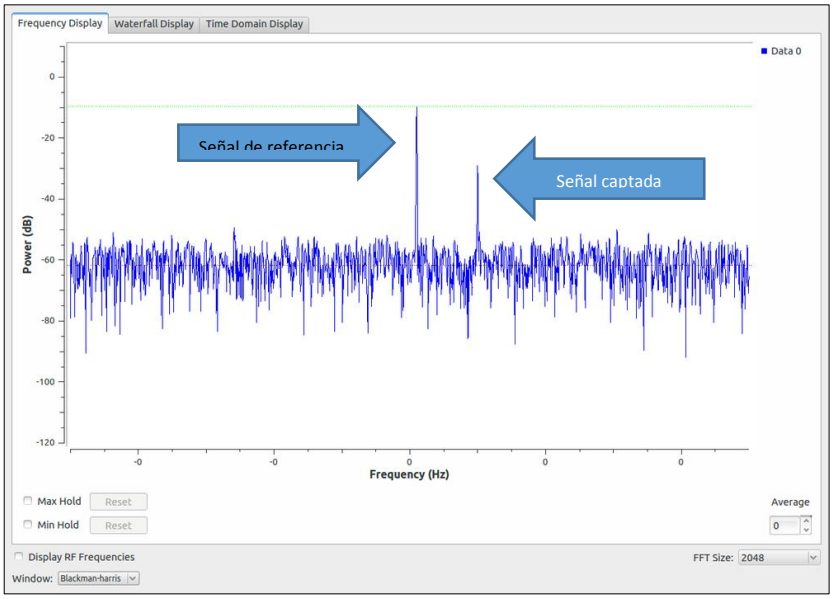


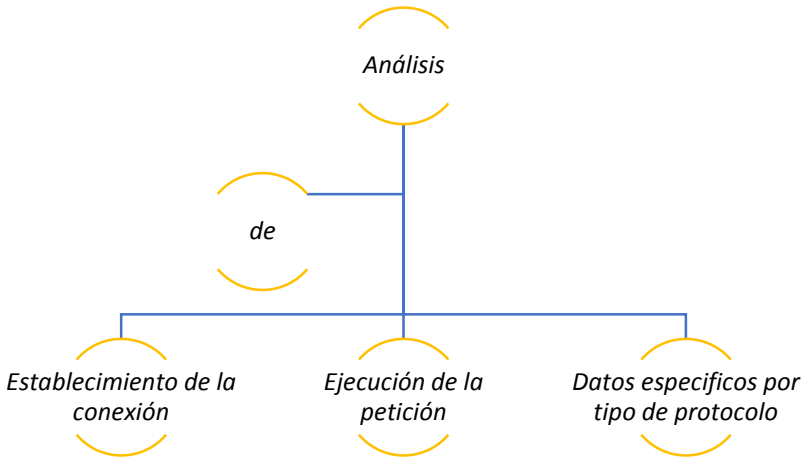
Figura 83 Bloque réplica de señal – propiedades

Fuente: Investigador

Al ejecutar el flujograma de réplica la segunda señal aparecerá periódicamente, este tiempo tiene relación al tiempo tomado en la fase de captación. Esta señal seguirá operando inclusive después de ejecutar satisfactoriamente su petición. Se recomienda dejar de correr el flujograma una vez realizada la orden.

4.7.6 Análisis de la pila de protocolos Wifi

Wifi estandarizada por el protocolo 802.11 define a la pila de protocolos TCP/IP como medio para trasmitir información en esta tecnología. Los dispositivos involucrados en esta comunicación deben escoger ciertos parámetros antes de poder establecerla. Este proceso permite establecer un esquema de análisis de datos que partirá desde:



Establecimiento de la conexión

El módulo Wi-Fi Esp8266 permite establecer comunicación con otro dispositivo inalámbrico mediante una dirección IP. El desarrollo de la aplicación (Wifi) motivo de Hacking ético es una página web programada con código HTML en el dispositivo inalámbrico el cual se conecta al equipo proveedor de internet; este le asigna una dirección IP, la misma que servirá para poder ingresar como url desde cualquier equipo conectado a la red y poder ejecutar las peticiones de encendido y apagado de luz remotamente.

- El equipo usado remotamente serán los posibles clientes
- El Modulo Esp8266 será el servidor que ejecutara las peticiones

Cada equipo conectado a la red tiene un número de identificación de 48 bits. Este es un número único establecido en el momento de la fabricación de la tarjeta. El protocolo de resolución de dirección tiene un papel clave entre los protocolos de capa de Internet relacionados con el protocolo TCP/IP, ya que permite que se conozca la dirección física de una tarjeta de interfaz de red correspondiente a una dirección IP.

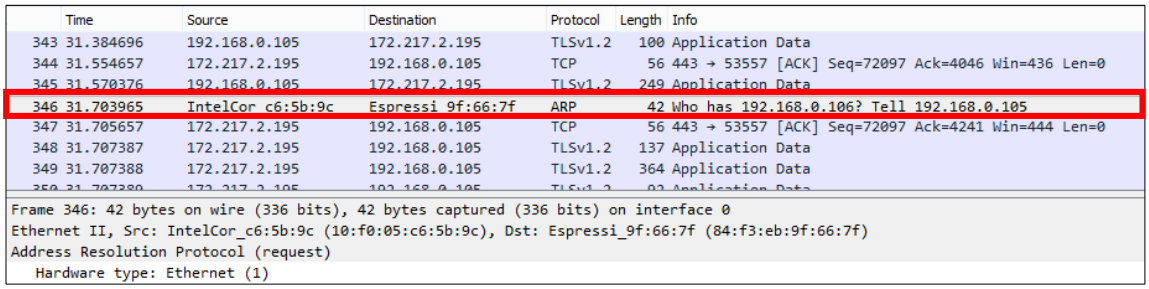


Figura 84 Protocolo ARP – asignación de direccione ip

Fuente: Investigador

- La trama 346 indicada en la figura 85 muestra el establecimiento de las direcciones IP mediante la dirección MAC de cada dispositivo. *Source.- IntelCor_C6: (10:f0:05:c6:5b:9c), Destination.- Aspire_9f:66:7f (84:f3:ed:9f:66:7f)*

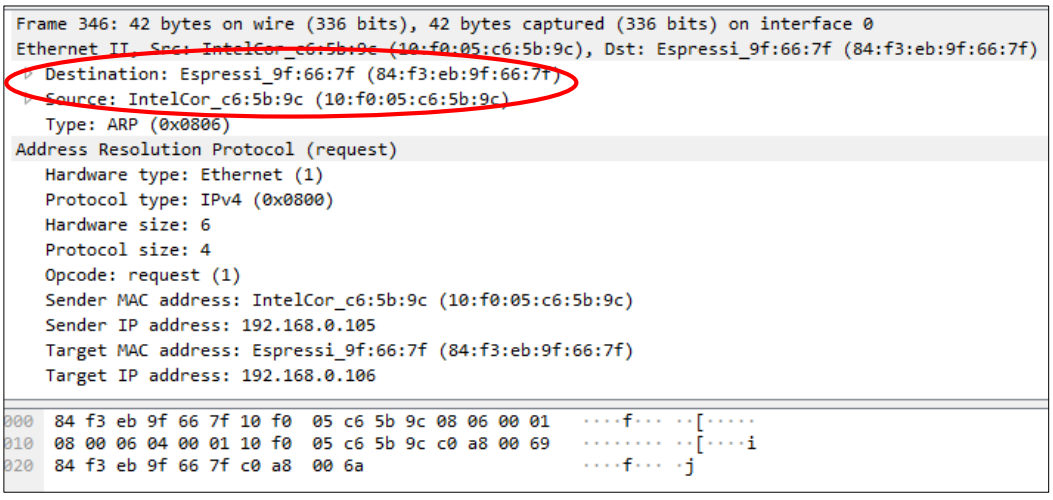


Figura 85 Protocolo ARP especificaciones

Fuente: Investigador

1.- Hardware type: Ethernet (1).- Este campo especifica el tipo de protocolo de enlace de red

2.- Protocol type: IPv4 (0x0800).- Este campo especifica el protocolo de internet para el que se pretende la solicitud ARP. Para IPv4, este tiene el valor 0x0800. Los valores de PTYPE permitidos comparten un espacio de numeración con los de EtherType.

3.- Hardware Size (6).- Longitud (en octetos) de una dirección de hardware. El tamaño de las direcciones Ethernet es 6.

4.- Protocol size (4).- Longitud (en octetos) de las direcciones utilizadas en el protocolo de capa superior. (El protocolo de capa superior especificado en PTYPE.) El tamaño de la dirección IPv4 es 4.

De igual forma al establecer un nivel de comunicación el dispositivo destino envía un acuse de recibo al dispositivo principal confirmando la asignación de direcciones y a la espera de una trasmisión de datos. La trama 355 indicada en la figura 87 muestra el proceso mencionado anteriormente.

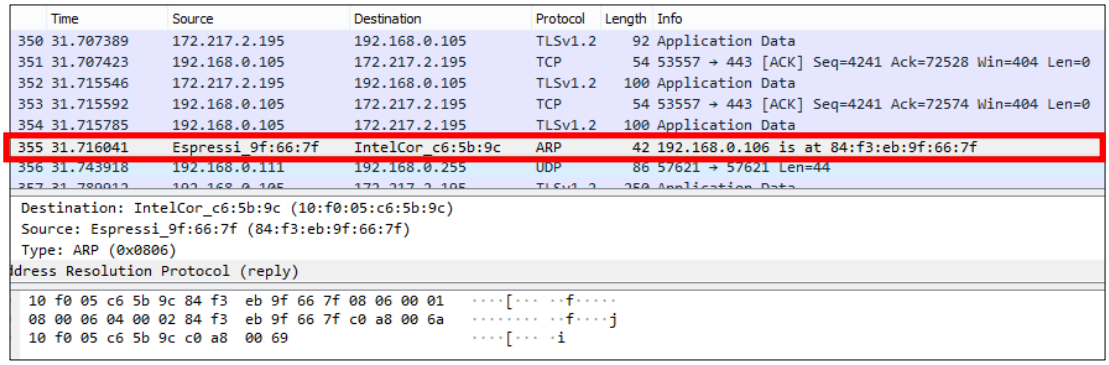
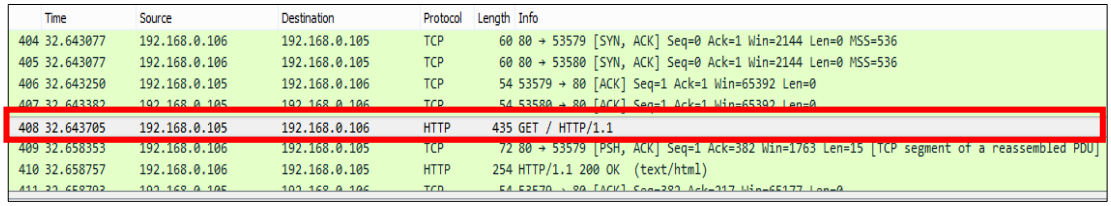


Figura 86 Acuse de recibo por parte del dispositivo remoto

Fuente: Investigador

Ejecución de la petición

Se parte de la base que el cliente realiza una petición al servidor requiriendo información de una página determinada. Esta petición tiene una respuesta como muestra la figura 89.



	Time	Source	Destination	Protocol	Length	Info
404	32.643077	192.168.0.106	192.168.0.105	TCP	60	80 → 53579 [SYN, ACK] Seq=0 Ack=1 Win=2144 Len=0 MSS=536
405	32.643077	192.168.0.106	192.168.0.105	TCP	60	80 → 53580 [SYN, ACK] Seq=0 Ack=1 Win=2144 Len=0 MSS=536
406	32.643250	192.168.0.105	192.168.0.106	TCP	54	53579 → 80 [ACK] Seq=1 Ack=1 Win=65392 Len=0
407	32.643382	192.168.0.105	192.168.0.106	TCP	54	53580 → 80 [ACK] Seq=1 Ack=1 Win=65392 Len=0
408	32.643705	192.168.0.105	192.168.0.106	HTTP	435	GET / HTTP/1.1
409	32.658353	192.168.0.106	192.168.0.105	TCP	72	80 → 53579 [PSH, ACK] Seq=1 Ack=382 Win=1763 Len=15 [TCP segment of a reassembled PDU]
410	32.658757	192.168.0.106	192.168.0.105	HTTP	254	HTTP/1.1 200 OK (text/html)
411	32.658793	192.168.0.105	192.168.0.106	TCP	54	53579 → 80 [ACK] Seq=382 Ack=217 Win=65177 Len=0

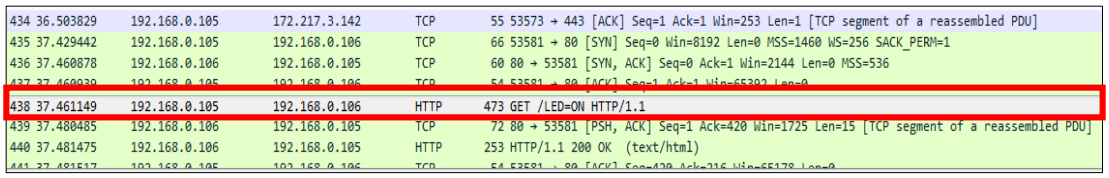
Figura 87 Ejecución de la petición

Fuente: Investigador

HTTP define un conjunto de métodos de petición para indicar la acción que se desea realizar para un recurso determinado. Estos métodos de solicitud a veces son llamados HTTP verbs. Como se observa en la figura anterior el método utilizado por este protocolo es el GET; solicita una representación de un recurso específico. Las peticiones que usan el método GET sólo deben recuperar datos. Esta respuesta tiene los protocolos TCP/IP en su composición, al estar solicitando una recuperación de datos muestra mediante le analizador de protocolos el método GET usado en la elaboración de la página web contenida en el servidor.

Datos específicos por tipo de protocolo

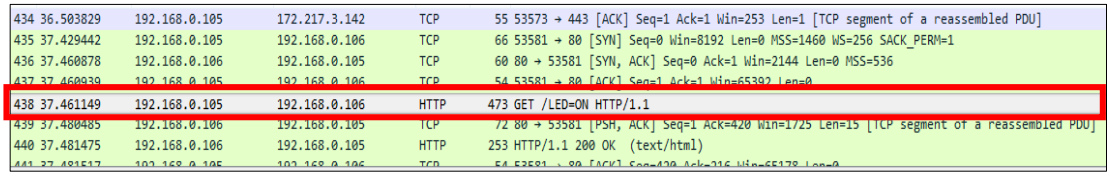
En el desarrollo del código HTML contenido en el servidor (Módulo Esp8266) si la estructura de requerimientos de cliente contiene la cadena *"/LED=ON"* el estado del pin estará en 1 lógico (3.3v) de lo contrario si la cadena es *"/LED=OFF"* el estado del pin estará en 0 lógico (0v). Las figuras 89 y 90 mostradas a continuación indican el proceso descrito anteriormente.



434	36.503829	192.168.0.105	172.217.3.142	TCP	55	53573 → 443 [ACK] Seq=1 Ack=1 Win=253 Len=1 [TCP segment of a reassembled PDU]
435	37.429442	192.168.0.105	192.168.0.106	TCP	66	53581 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
436	37.460878	192.168.0.106	192.168.0.105	TCP	60	80 → 53581 [SYN, ACK] Seq=0 Ack=1 Win=2144 Len=0 MSS=536
437	37.460939	192.168.0.105	192.168.0.106	TCP	54	53581 → 80 [ACK] Seq=1 Ack=1 Win=65392 Len=0
438	37.461149	192.168.0.105	192.168.0.106	HTTP	473	GET /LED=ON HTTP/1.1
439	37.480485	192.168.0.106	192.168.0.105	TCP	72	80 → 53581 [PSH, ACK] Seq=1 Ack=420 Win=1725 Len=15 [TCP segment of a reassembled PDU]
440	37.481475	192.168.0.106	192.168.0.105	HTTP	253	HTTP/1.1 200 OK (text/html)
441	37.481517	192.168.0.105	192.168.0.106	TCP	54	53581 → 80 [ACK] Seq=420 Ack=216 Win=65178 Len=0

Figura 88 Establecimiento de la conexión – Pin en modo activo

Fuente: Investigador



434	36.503829	192.168.0.105	172.217.3.142	TCP	55	53573 → 443 [ACK] Seq=1 Ack=1 Win=253 Len=1 [TCP segment of a reassembled PDU]
435	37.429442	192.168.0.105	192.168.0.106	TCP	66	53581 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
436	37.460878	192.168.0.106	192.168.0.105	TCP	60	80 → 53581 [SYN, ACK] Seq=0 Ack=1 Win=2144 Len=0 MSS=536
437	37.460939	192.168.0.105	192.168.0.106	TCP	54	53581 → 80 [ACK] Seq=1 Ack=1 Win=65392 Len=0
438	37.461149	192.168.0.105	192.168.0.106	HTTP	473	GET /LED=ON HTTP/1.1
439	37.480485	192.168.0.106	192.168.0.105	TCP	72	80 → 53581 [PSH, ACK] Seq=1 Ack=420 Win=1725 Len=15 [TCP segment of a reassembled PDU]
440	37.481475	192.168.0.106	192.168.0.105	HTTP	253	HTTP/1.1 200 OK (text/html)
441	37.481517	192.168.0.105	192.168.0.106	TCP	54	53581 → 80 [ACK] Seq=420 Ack=216 Win=65178 Len=0

Figura 89 Establecimiento de la conexión – Pin en modo pasivo

Fuente: Investigador

Esta ejecución de petición es una interacción entre cliente servidor y a diferencia de otra tecnologías como bluetooth cubre toda la pila de protocolos. A continuación se muestra en la figura 91 un análisis amplio de la pila TCP/IP en la aplicación desarrollada

Figura 90 Pila de protocolos TCP/IP

Fuente: Investigador

ETHERNET II

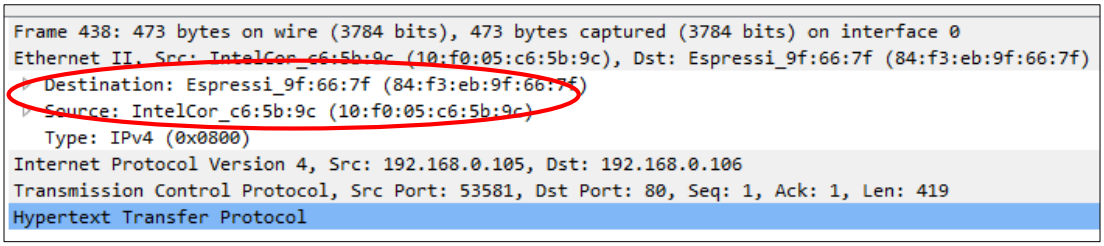


Figura 91 Protocolo de acceso a la red – Ethernet

Fuente: Investigador

- *Source: IntelCor_c6:5b:9c (10:f0:05:c6:5b:9c).*- Indica la dirección Mac del servidor (módulo Esp8622).
- *Destination: Espressi_9f:66:7f (10:f0:05:c6:5b:9c).*- Muestra la dirección Mac del dispositivo remoto gestor de las peticiones.
- *Type: IPv4 (0x0800).*- Indica el tipo de direccionamiento (versión 4).

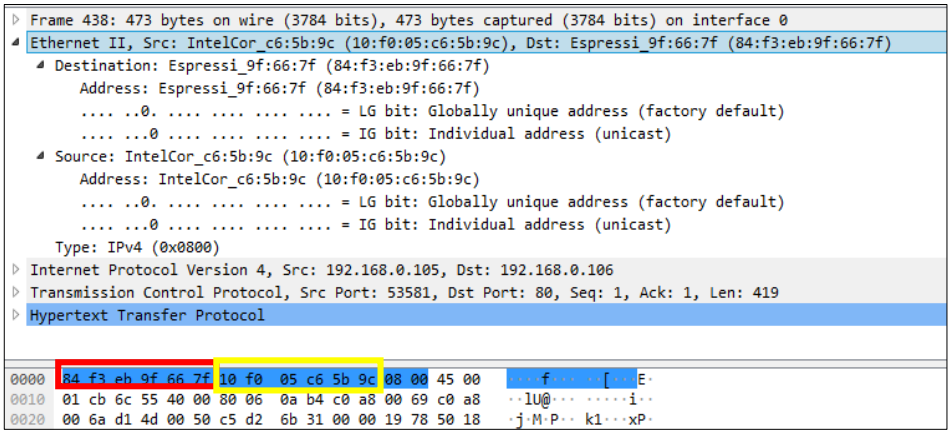


Figura 92 Bytes del encabezado Ethernet

Fuente: Investigador

- La figura 93 muestra los 14 bytes del encabezado Ethernet .Los primeros 6 octetos corresponden a su dirección destino los 6 posteriores corresponden a su dirección origen y los dos últimos al direccionamiento.

Internet Protocol

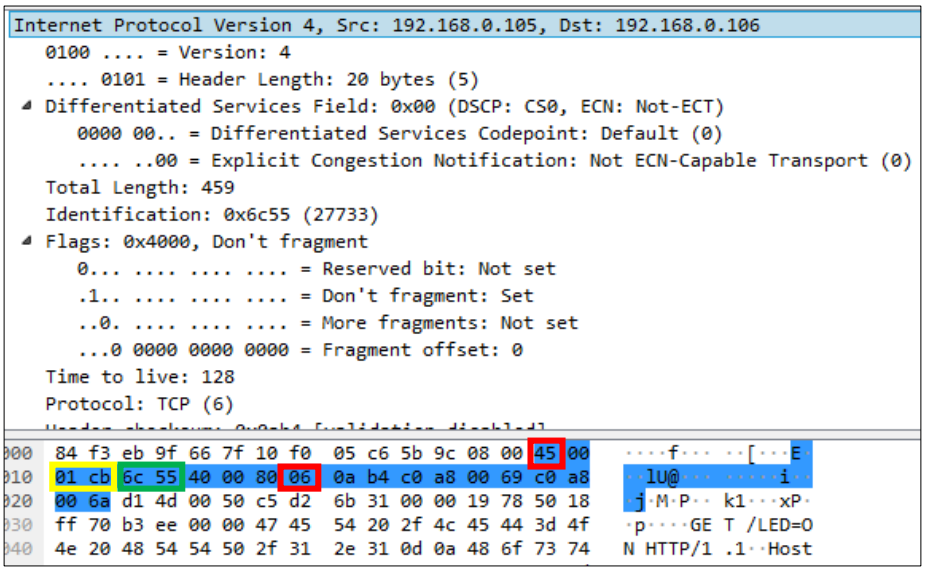


Figura 93 Protocolo de internet

Fuente: Investigador

- *Versión: 4: 0100… .-* Indica la versión de direccionamiento IP (versión 4)
- *Header lenght: 20 bytes: ....0101 .-* Muestra la cabecera de la capa de internet con sus 20 bytes de encabezado

- *Differentatied service field*.- Es una identificación única aplicada a cada paquete que un host envía en una conexión particular. En general, solo es útil si un paquete necesita fragmentarse (por ejemplo, mediante un enrutador): cada fragmento conservará la identificación original. Permite que el host receptor sepa cómo volver a armar los fragmentos.
- *Total lenght: 49*.- Representa la longitud de la trama.
- *Identification: 0x6c55*.- Muestra el número de identificación de trama (nivel 2)
- *Protocol: TCP (6)*.- Sirve para identificar el siguiente nivel de protocolo el cual se denomina encabezado posterior.
- Los bytes sobrantes representan a las direcciones IP origen y destino de los elementos involucrados en la comunicación.

Transmisión control Protocol

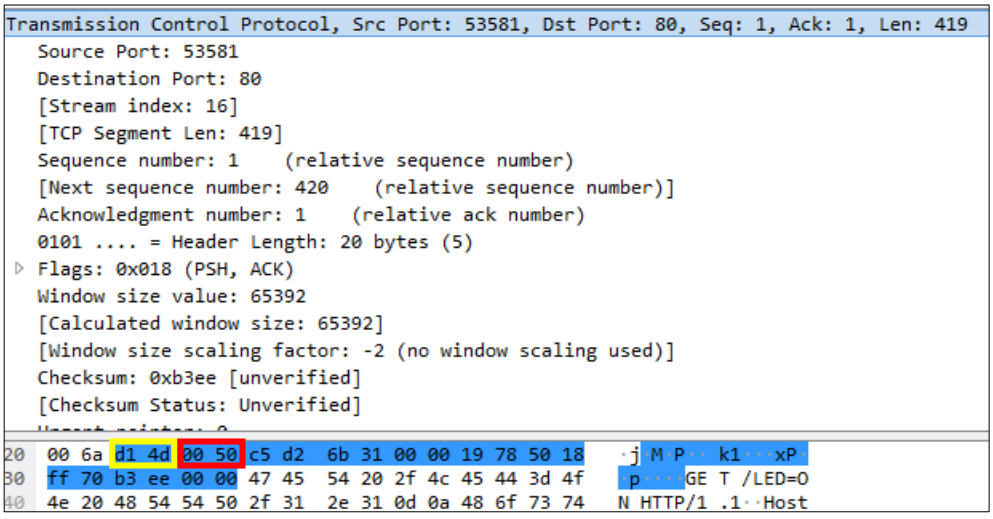


Figura 94 Protocolo de trasmisión

Fuente: Investigador

- *Source Port:53581*.- Es el puerto del cliente por donde se envia el pedido al servidor http.
- *Destination Port: 80* .- Es que puerto por default, por el medio del cual un servidor HTTP "escucha" la petición hecha por un cliente, es decir por una PC en específico.
- *Header lenght 20 bytes: 0101 ....* .- Indica la cabecera de la capa de trasporte con sus 20 bytes de encabezado.

Hypertext transfer Protocol

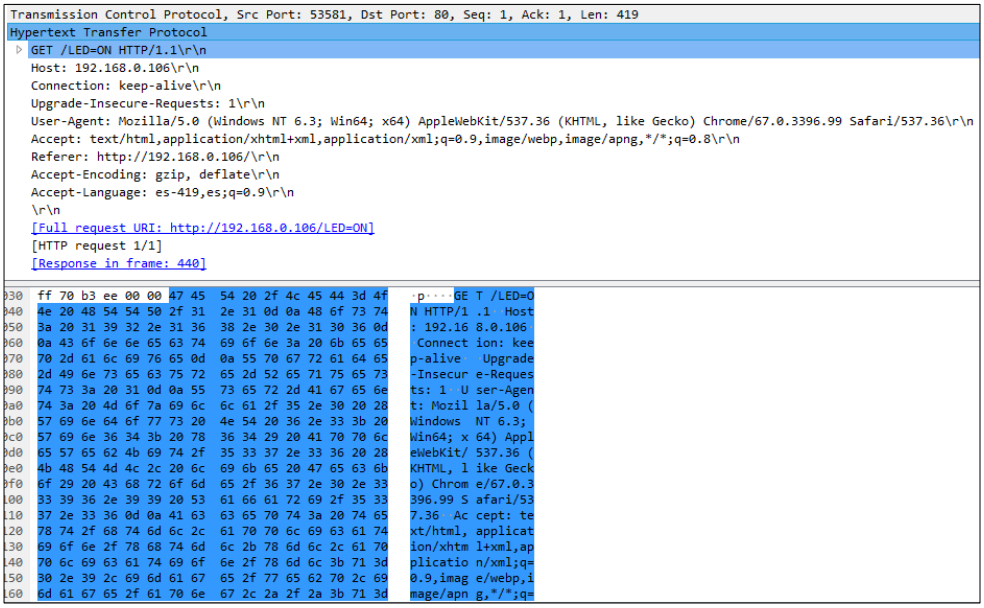


Figura 95 Protocolo de Aplicación

Fuente: Investigador

- *Host: 192.168.0.106.*- Indica la dirección IP del host donde se realiza la petición al servidor.
- Este protocolo indica el tipo de navegador usado para el establecimiento de la conexión, sistema operativo del host y como característica principal el estado del elemento final (on – off).

4.8 Modelo de arquitectura de sistemas de comunicaciones orientados al IoT para el desarrollo de hacking ético mediante SDR

Una vez realizado el proyecto de investigación, tomando en cuenta el diseño de aplicaciones para tecnologías IoT (Bluetooth, Wifi, Z wave) para la aplicación de un hacking ético, se propone el siguiente modelo de arquitectura.

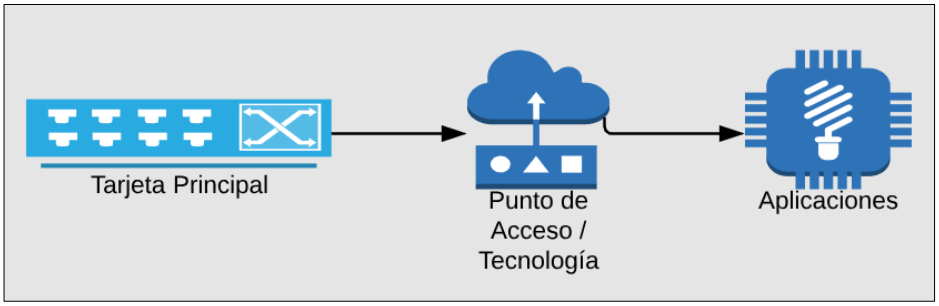


Figura 96 Modelo de arquitectura de sistemas de comunicaciones orientados al IoT para el desarrollo de hacking ético mediante SDR

Fuente: Investigador

A continuación se detalla la arquitectura propuesta como indica la figura 96:

Tarjeta Principal.- Dentro de este bloque se encuentra a los sensores, actuadores y el hardware necesario (dispositivo principal: Arduino) para comunicar el mundo físico con el mundo virtual. Actualmente existe infinidad de componentes, sobre todo para el prototipado, que permiten comenzar a crear dispositivos del IoT.

Puntos de acceso.- Los puntos de acceso permiten la conectividad de las cosas, objetos o dispositivos a Internet. El objetivo fundamental es establecer una conexión entre los periféricos (dispositivos u objetos conectados) y la nube o tecnología, permitiendo la conectividad entre ellos. Esta conexión tiene que ser segura, robusta, tolerante a fallos a fin de que recoja la información obtenida de los dispositivos y a la vez, se puedan gestionar.

Aplicaciones.- Las aplicaciones sirven para poder manejar y visualizar la información, da lo mismo si son nativas o web. Gracias al uso de APIs y servicios web, cualquier tipo de aplicación se podrá conectar a los datos y mostrarlos a los usuarios. Además de visualizar los datos las aplicaciones tendrá la capacidad de modificar los parámetros para que los sistemas se comporten de una manera determinada.

4.9 Detalle de vulnerabilidades de seguridad encontradas en tecnologías IoT.

Una vez realizado el análisis de la trama obtenida en el proceso de hacking ético se presenta un detalle de vulnerabilidades encontradas en la tecnología bluetooth y wifi en cada uno de los protocolos indicados anteriormente (ver punto 4.7).

Tabla 10 Vulnerabilidades de seguridad a nivel de protocolo

Vulnerabilidades de seguridad a nivel de protocolo			
	Tecnología Bluetooth		*Tecnología Wi_Fi*
Protocolo HCI	✓ Se irrumpe la interfaz de control de Host, es ahí donde se observa la dirección MAC del dispositivo maestro - esclavo en la trama Bluetooth capturada.	***Capa de Red***	✓ Irrupción al control de acceso y confidencialidad. Se indica las direcciones MAC de los dispositivos involucrados en la comunicación
L2CAP	✓ Se determina el orden en los canales de frecuencia pasando por alto la seguridad en el salto de canal FHSS origen - destino. ✓ Como solo se realiza una identificación de extremos (acuse de recibo), cualquiera de ellos podría ser suplantado y se podría continuar con la comunicación**.**	***Capa de Red***	✓ Fallos en la seguridad del datagrama IP. ✓ Se puede realizar sniffing y modificación de mensajes
RFCOM	✓ Se expone claramente los parámetros del protocolo de transporte: acceso, cabecera, payload. Aunque no indica todos los bits que lo componen (rfcom), si indica las cabeceras y bits principales de información.	***Capa de transporte***	✓ Se puede verificar autenticación, de integridad y confidencialidad con esto se puede realizar una réplica de datos.
SPP	✓ Indica el dato capturado sin ningún tipo encriptación.	***Capa de Aplicación***	✓ Se muestra parámetros de construcción en HTPP.

Nota: El análisis de seguridad a nivel de protocolo solo se presenta tanto de bluetooth como de wifi, la tecnología Z wave no se indica puesto que la información enviada va libre y no encriptada de este modo no existe un nivel de seguridad informática que presente vulnerabilidades.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- En el desarrollo de sistemas de comunicaciones mediante diagramas de flujo uno de los requisitos principales para su diseño es la frecuencia de muestreo. Para las tres tecnologías inalámbricas orientadas al IoT esta frecuencia está en el rango de los 20 MHz, este parámetro es aún más importante en la captación de señal de bloqueo o desbloqueo en la seguridad vehicular de la tecnología Z wave puesto que el pulso inicial del mando del control debe ser captado por la antena en su totalidad, que de no estar en su valor correcto y al concurrir con pulsos periódicos la señal captada sufrirá solapamiento y la ejecución de la petición no lograra su fin.
- La tecnología Z wave en el modo unautenticated permite realizar réplicas de señal debido a que no consta de encriptación, la señal es tipo pura por lo que no necesita bloques de modulación, selección de canales de trasmisión y arreglos de señales adicional, etc. Se puede tomar como condición que la ganancia del segundo flujograma (replica) se duplique de 20dBm a 40 dBm por motivos de amplificación además de una consideración de distancia por motivos de efectividad al ejecutar la acción.
- En el análisis de datos obtenidos en la tecnología bluetooth se puede ver una trama compuesta por 15 bytes (120 bits) que representan los protocolos hci, l2cap, rfcom y spp capturados. Los demás protocolos involucrados en la tecnología no mostrados a partir de HCI de manera ascendente son aquellos considerados como direccionamiento de paquete que no son descifrados por el analizador, esto se pudo determinar mediante el formato Little Endian, donde bit menos significativo (LSB) corresponde a b0 mostrado en la trama de establecimiento de la conexión, este bit es el primero en enviarse y con esto determinar que tanto el código de acceso, cabecera y payload solo muestran sus encabezados y datos enviados respectivamente.
- El diseño del diagrama de flujo para captar la señal Wifi, tiene como bloque principal al demodulador, donde se ajusta todas las características posibles

conocidas de la señal captada, no es necesario conocer el canal de comunicación debido a que casi siempre la asignación dinámica de canal por defecto es el 11, a este demodulador se le asigna un filtro pasa bajo para depurar las señales en el ambiente, según el trabajo desarrollado la presentación de la señal en modo constelación es un factor importante conocer el estado de la señal captada.

5.2 Recomendaciones

- Es de utilidad instalar una interfaz asociada a SDR llamada gqrx que permite mediante diagramas de señal mostrar si el equipo está actuando como captador de señal y si la frecuencia de muestreo está en el rango correcto, esto se puede verificar cuando al ejecutar una orden se muestra en la pantalla pulsos de señal como motivo de verificación.
- Para la fase uno en la tecnología Z wave es recomendable acortar la distancia entre el dispositivo principal el equipo receptor por motivos de efectividad, así mismo la ejecución debe ser realizada el número menor posible para tener una señal más limpia y sin solapamiento.
- En el análisis de los datos de la tecnología bluetooth solo se tomó en consideración los encabezados de cada protocolo, el analizador no muestra los 1645 bits, por lo que se recomienda conocer el número de bytes de encabezado de cada protocolo que compone esta tecnología.
- Al aplicar el diagrama de hacking a la tecnología Wifi, tomando en consideración el punto anterior el diseño del autor cumple con las objetivos expuestos sin embargo la recepción se la debe hacer en el menor tiempo posible por efectos de calentamiento y consumo casi en su totalidad al equipo de cómputo, tratando de evitar algún colapso.

BIBLIOGRAFÍA

[1] D. Evans, «Cisco,» 15 Julio 2011. [En línea]. Available: https://www.cisco.com/c/dam/global/es_mx/solutions/executive/assets/pdf/internet-of-things-iot-ibsg.pdf. [Último acceso: 5 Diciembre 2017].

[2] M. S. a. T. Beardsley, «Rapid7,» 29 Septiembre 2015. [En línea]. Available: https://www.rapid7.com/docs/Hacking-IoT-A-Case-Study-on-Baby-Monitor-Exposures-and-Vulnerabilities.pdf. [Último acceso: 5 Diciembre 2017].

[3] J. M. -. J. Godon, «Hacking the IoT: when good devices go bad,» 4 Marzo 2016. [En línea]. Available: https://www.rsaconference.com/writable/presentations/file_upload/sbx1-w07-when-good-devices-go-bad-live-hacking-in-the-iot.pdf . [Último acceso: 5 Diciembre 2017].

[4] J. L. Ziegler, R. T. Arn y W. Chambers, «Modulation recognition with GNU radio, keras and HackRF,» 12 Julio 2015. [En línea]. Available: http://ieeexplore.ieee.org/document/7920747/. [Último acceso: 5 Diciembre 2017].

[5] D. Vohra, A. Dubey y K. Vachhhani, «IEEE,» 15 Mayo 2016. [En línea]. Available: http://ieeexplore.ieee.org/document/7566288/. [Último acceso: 8 Diciembre 2017].

[6] c. System, «Lo que usted necesita saber de redes inalambricas,» 25 Agosto 2012. [En línea]. Available: https://www.cisco.com/c/dam/global/es_mx/assets/ofertas/desconectadosanonimos/wireless/pdfs/brochure_wireless.pdf. [Último acceso: 6 Diciembre 2017].

[7] C. Luis, «li-fi,» [En línea]. Available: http://lanuevaredli-fi.blogspot.com/2016/01/evolucion-de-las-redes-inalambricas.html. [Último acceso: 5 Diciembre 2017].

[8] J. Torrez, «¿Qué es y cómo funciona el Internet de las cosas?,» Hipertextual, 14 Octubre 2014. [En línea]. Available: https://hipertextual.com/archivo/2014/10/internet-cosas/. [Último acceso: 6 Diciembre 2017].

[9] L. d. V. Hernández, «Arquitectura IoT, prototipando los dispositivos del futuro,» ProgramarFacil, 22 Enero 2015. [En línea]. Available: https://programarfacil.com/podcast/arduino-wifi-proyectos-iot/. [Último acceso: 6 Diciembre 2017].

[10] D. R. González, «Revista digital de tecnologias de la información,» 13 Septiembre 2013. [En línea]. Available: http://revistatelematica.cujae.edu.cu/index.php/tele/article/viewFile/119/115. [Último acceso: 6 Diciembre 2017].

[11] E. Rico, «Ermesh,» 19 Abril 2015. [En línea]. Available: http://www.ermesh.com/modelos-de-comunicacion-para-internet-de-las-cosas/. [Último acceso: 7 Diciembre 2017].

[12] A. B. Beltrán, «Arquitectura de referecnia en el IoT,» *SUNQU*, vol. I, nº 11, pp. 15-24, 2016.

[13] A. J. F. González, «Dispositivos IOT, la red de las cosas, protocolos de comunicacion,» beBee, 7 Julio 2017. [En línea]. Available: https://www.bebee.com/producer/@http-alfrdo-j-feijoo-webnode-es-contacto/dispositivos-iot-la-red-de-las-cosas-protocolos-de-comunicacion. [Último acceso: 8 Diciembre 2017].

[14] «Seguridad integrada y en la nube para el Internet de las Cosas,» Gemalto, 6 Abril 2016. [En línea]. Available: https://www.gemalto.com/latam/iot/seguridad-en-iot. [Último acceso: 9 Diciembre 2017].

[15] L. F. d. l. Mora, «Oficina de seguridad del Internauta,» OSI, 12 Diciembre 2016. [En línea]. Available: https://www.osi.es/es/actualidad/blog/2017/10/02/la-ciberseguridad-es-una-

responsabilidad-de-todos-el-iot-y-sus-riesgos. [Último acceso: 9 Diciembre 2017].

[16] Anónimo, «Que es un protocolo,» Culturacion, 14 Febrero 2010. [En línea]. Available: http://culturacion.com/que-es-un-protocolo-de-red/. [Último acceso: 10 Diciembre 2017].

[17] Arrow, «Protocolos para la Internet de las cosas,» Arrow, 16 Junio 2015. [En línea]. Available: https://www.arrow.com/es-mx/research-and-events/articles/protocols-for-the-internet-of-things. [Último acceso: 10 Diciembre 2017].

[18] E. Research, «GNU Radio,» National Instrument company, 12 Diciembre 2017. [En línea]. Available: https://www.ettus.com/sdr-software/detail/gnu-radio. [Último acceso: 12 Diciembre 2017].

[19] J. J. M. F. Iván Pinar Domínguez, «Laboratorio de comunicaciones Digitales SDR,» *TSC*, vol. I, nº 1, p. 37, 2011.

[20] A. G. Quintero, «RoboticsLab,» 23 Julio 2016. [En línea]. Available: http://roboticslab.uc3m.es/publications/Articulo1.pdf. [Último acceso: 12 Diciembre 2017].

[21] L. R. Albert S. Huang, Bluetooth Essentials for Programmers, Bon: OSX, 2012.

[22] C. S. Alberto, «Tecnológia Bluetooth,» Editorial Privada, Mexico, 2008.

[23] O. B, «Kerchak,» 12 Enero 2015. [En línea]. Available: https://kerchak.com/ventajas-de-la-tecnologia-bluetooth/. [Último acceso: 13 Diciembre 2017].

[24] A. Ruiz, «Techlandia,» 13 Agosto 2011. [En línea]. Available: https://techlandia.com/ventajas-desventajas-del-bluetooth-lista_516646/. [Último acceso: 15 Diciembre 2017].

[25] A. M. Tablado, «Seguridad en bluetooth,» Editorial Madrid, Madrid, 2006.

[26] R. a. Schwazar, «Rodhe and Schwazar,» 12 Abril 2015. [En línea]. Available: https://www.rohde-schwarz.com/es/tecnologias/conectividad-inalambrica/zigbee/tecnologia-zigbee/tecnologia-zigbee_55724.html. [Último acceso: 14 Diciembre 2017].

[27] A6, «Tecnología Zigbee,» 19 Enero 2013. [En línea]. Available: http://www.ptolomeo.unam.mx:8080/jspui/bitstream/132.248.52.100/229/6/A6.pdf. [Último acceso: 14 Diciembre 2017].

[28] J. M. Moreno, «Informe Técnico: Protocolo ZigBee (IEEE 802.15.4),» 17 Agosto 2015. [En línea]. Available: https://rua.ua.es/dspace/bitstream/10045/1109/7/Informe_ZigBee.pdf . [Último acceso: 14 Enero 2017].

[29] M. P. Peña, «SCRIBD,» 4 Abril 2011. [En línea]. Available: https://es.scribd.com/document/52276185/TECNOLOGIA-INALAMBRICA-ZIGBEE. [Último acceso: 15 Diciembre 2017].

[30] 7GGraus, «Significados,» 12 Septiembre 2015. [En línea]. Available: https://www.significados.com/wifi/. [Último acceso: 15 Diciembre 2017].

[31] C. Martinoli, «WI-FI,» Editorial Mexico, Mexico, 2014.

[32] J. A. J. A. C. F. Carballar, Wi-fi lo que necesitas saber., Madrid: Grupo Ramirez cogollor, 2010.

[33] L. Rivera, «Topologias de una red WLAN,» Publicaciones ANVAL, Lima, 2015.

[34] G. Carlos, «KIOSERA,» 20 MArzo 2017. [En línea]. Available: https://smarterworkspaces.kyocera.es/blog/tipos-de-seguridad-wifi-mas-habituales/. [Último acceso: 16 Diciembre 2017].

[35] R. Vega, «Blog,» 16 Mayo 2016. [En línea]. Available: https://ricveal.com/blog/z-wave/. [Último acceso: 17 Diciembre 2017].

[36] PHILIPE, «Domótica Doméstica,» 23 Enero 2015. [En línea]. Available: http://www.domoticadomestica.com/asociaciones-directas-en-z-wave/. [Último acceso: 17 Diciembre1 2017].

[37] M. Mancuz, «Blog Domotica,» 9 Octubre 2017. [En línea]. Available: https://blog.domoticalia.es/2017/10/09/seguridad-iot-tecnologia-inalambrica-z-wave/. [Último acceso: 17 Diciembre 2017].

[38] n. plata, «Comunicaciones PC,» 15 Enero 2011. [En línea]. Available: https://comunicacionesupc.wordpress.com/2011/01/15/bandas-de-frecuencias-no-licenciadas/. [Último acceso: 17 Diciembre 2017].

[39] M. Osman, «GreatScottgadgets,» 16 Septiembre 2015. [En línea]. Available: https://greatscottgadgets.com/hackrf/. [Último acceso: 19 Diciembre 2017].

ANEXOS

Anexo A.- Código de programación en ide de Arduino _ Aplicación Bluetooth

```
//Declaración del pin 13 para activar y desactivar la luz
int led13=13;
int estado=0;
void setup() {
Serial.begin(9600);
pinMode(led13,OUTPUT);
}
//Asignación de los símbolos para generar peticiones remotas 1
      encendido 2 apagado
void loop() {
if(Serial.available()>0){
estado= Serial.read();
Serial.print(estado);
}
if (estado =='1'){
digitalWrite(led13,HIGH);
}
if (estado== '2'){
digitalWrite(led13,LOW);
                }
}
```

Anexo B.- Código de programación en ide de Arduino _ Aplicación Wifi

```
//inclucion de la librería del módulo esp8266
#include <ESP8266WiFi.h>
const char* ssid = "RED POL";
const char* password = "pelileo21";
int ledPin = 13; // GPIO13
WiFiServer server(80);
void setup() {
Serial.begin(115200);
delay(10);
pinMode(ledPin, OUTPUT);
digitalWrite(ledPin, LOW);
   // Conexión a una red wifi
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
   // Iniciar el servidor
  server.begin();
  Serial.println("Server started");
   // Imprimir la dirección ip
  Serial.print("Use this URL to connect: ");
```

```
 Serial.print("http://");
 Serial.print(WiFi.localIP());
 Serial.println("/");
}
void loop() {
 // Check if a client has connected
 WiFiClient client = server.available();
 if (!client) {
   return;
 }
  // Tiempo de espera para recibir un dato
 Serial.println("new client");
 while(!client.available()){
   delay(1);
 }
  // Lectura de la primera línea de respuesta
 String request = client.readStringUntil('\r');
 Serial.println(request);
 client.flush();
  // Enlace de respuesta
  int value = LOW;
 if (request.indexOf("/LED=ON") != -1)  {
   digitalWrite(ledPin, HIGH);
   value = HIGH;
 }
 if (request.indexOf("/LED=OFF") != -1)  {
   digitalWrite(ledPin, LOW);
   value = LOW;
 }
```

```
 // envio de ledPin de acuerdo a la respuesta

//digitalWrite(ledPin, value);

   // Return the response

  client.println("HTTP/1.1 200 OK");

  client.println("Content-Type: text/html");

  client.println(""); //  do not forget this one

  client.println("<!DOCTYPE HTML>");

  client.println("<html>");

   client.print("Led pin is now: ");

   if(value == HIGH) {

    client.print("On");

  } else {

    client.print("Off");

  }

  client.println("<br><br>");

  client.println("<a href=\"/LED=ON\"\"><button>Turn On
      </button></a>");

  client.println("<a href=\"/LED=OFF\"\"><button>Turn Off
      </button></a><br />");

  client.println("</html>");

   delay(1);

  Serial.println("Client disonnected");

  Serial.println("");

 }
```

Anexo C.- Código de programación en eclipse _ formulario de presentación de datos.

// Menú principal

```
package tesis;

import java.awt.BorderLayout;

import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.border.EmptyBorder;

import javax.swing.JLabel;

import java.awt.Color;

import java.awt.Font;

import javax.swing.JButton;

import javax.swing.ImageIcon;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;


public class frm_menu extends JFrame {

	private JPanel contentPane;

	/**

	 * Launch the application.

	 */

	public static void main(String[] args) {

		EventQueue.invokeLater(new Runnable() {

			public void run() {

				try {

					frm_menu frame = new frm_menu();

					frame.setVisible(true);
```

```
                } catch (Exception e) {
                        e.printStackTrace();
                }
            }
      });
}

/**
 * Create the frame.
 */
public frm_menu() {
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setBounds(100, 100, 566, 385);
      contentPane = new JPanel();
      contentPane.setBackground(new Color(0, 102, 153));
      contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
      setContentPane(contentPane);
      contentPane.setLayout(null);

      JLabel lblNewLabel = new JLabel("HACKING \u00C9TICO
MEDIANTE SDR");
      lblNewLabel.setFont(new Font("Comic Sans MS", Font.BOLD,
18));
      lblNewLabel.setForeground(Color.WHITE);
      lblNewLabel.setBounds(133, 11, 318, 65);
      contentPane.add(lblNewLabel);

      JButton btnNewButton = new JButton("");
      btnNewButton.addMouseListener(new MouseAdapter() {
            @Override
```

```
            public void mouseClicked(MouseEvent e) {
                  frm_wifi fr = new frm_wifi();
                  fr.setVisible(true);
            }
      });
      btnNewButton.setIcon(new ImageIcon("C:\\Users\\Andres
Valencia\\eclipse-workspace\\tesis\\img\\wifi.png"));
      btnNewButton.setBounds(223, 141, 89, 82);
      contentPane.add(btnNewButton);

      JButton btnBlue = new JButton("");
      btnBlue.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                  frm_bluet fr = new  frm_bluet();
                  fr.setVisible(true);
                  fr.setVisible(true);
            }

      });
      btnBlue.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
            }
      });
      btnBlue.setIcon(new ImageIcon("C:\\Users\\Andres
Valencia\\eclipse-workspace\\tesis\\img\\bluetooth.png"));
      btnBlue.setBounds(59, 141, 89, 82);
      contentPane.add(btnBlue);

      JButton button = new JButton("");
```

```
		button.addMouseListener(new MouseAdapter() {
			@Override
			public void mouseClicked(MouseEvent e) {
				frm_zwave fr = new  frm_zwave();
				fr.setVisible(true);

			}
		});
		button.setIcon(new ImageIcon("C:\\Users\\Andres Valencia\\eclipse-workspace\\tesis\\img\\car-key.png"));
		button.setBounds(389, 141, 89, 82);
		contentPane.add(button);

		JLabel lblBluetooth = new JLabel("Bluetooth");
		lblBluetooth.setFont(new Font("Comic Sans MS", Font.BOLD, 13));
		lblBluetooth.setBackground(Color.WHITE);
		lblBluetooth.setForeground(Color.WHITE);
		lblBluetooth.setBounds(69, 228, 76, 14);
		contentPane.add(lblBluetooth);

		JLabel lblWifi = new JLabel("Wi_Fi");
		lblWifi.setForeground(Color.WHITE);
		lblWifi.setFont(new Font("Comic Sans MS", Font.BOLD, 13));
		lblWifi.setBackground(Color.WHITE);
		lblWifi.setBounds(248, 228, 76, 14);
		contentPane.add(lblWifi);

		JLabel lblZwave = new JLabel("Z_Wave");
		lblZwave.setForeground(Color.WHITE);
```

```
            lblZwave.setFont(new Font("Comic Sans MS", Font.BOLD,
      13));
            lblZwave.setBackground(Color.WHITE);
            lblZwave.setBounds(410, 228, 76, 14);
            contentPane.add(lblZwave);
      }
}

// Formulario Wifi
package tesis;
import java.awt.BorderLayout;
import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import java.awt.Color;
import javax.swing.JLabel;
import java.awt.Font;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.ImageIcon;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
public class frm_wifi extends JFrame {
      private JPanel contentPane;
      /**
       * Launch the application.
       */
      public static void main(String[] args) {
            EventQueue.invokeLater(new Runnable() {
                  public void run() {
                        try {
```

```
                        frm_wifi frame = new frm_wifi();
                        frame.setVisible(true);
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            });
        }

        /**
         * Create the frame.
         */
        public frm_wifi() {
            addWindowListener(new WindowAdapter() {
                @Override
                public void windowOpened(WindowEvent arg0) {
                    panel1.setVisible(false);
                }
            });
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setBounds(100, 100, 564, 385);
            contentPane = new JPanel();
            contentPane.setBackground(new Color(0, 102, 153));
            contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
            setContentPane(contentPane);
            contentPane.setLayout(null);
            JLabel lblBluetooth = new JLabel("Wi_Fi");
            lblBluetooth.setBackground(new Color(0, 102, 153));
            lblBluetooth.setForeground(Color.WHITE);
            lblBluetooth.setFont(new Font("Comic Sans MS",
Font.BOLD, 35));
            lblBluetooth.setBounds(220, 11, 318, 65);
            contentPane.add(lblBluetooth);
                        JButton button = new JButton("");
            button.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
```

```
		}
	});
	button.addMouseListener(new MouseAdapter() {
		@Override
		public void mouseClicked(MouseEvent arg0) {
			panel1.setVisible(true);
		}
	});

	button.setIcon(new ImageIcon("C:\\Users\\Andres
Valencia\\eclipse-workspace\\tesis\\img\\newspaper.png"));
	button.setBounds(111, 96, 95, 78);
	contentPane.add(button);
	JLabel lblReporte = new JLabel("Reporte");
	lblReporte.setForeground(Color.WHITE);
	lblReporte.setFont(new Font("Comic Sans MS", Font.BOLD,
13));
	lblReporte.setBackground(Color.WHITE);
	lblReporte.setBounds(140, 179, 76, 23);
	contentPane.add(lblReporte);
	JButton btnNewButton = new JButton("Back");
	btnNewButton.addMouseListener(new MouseAdapter() {
		@Override
		public void mouseClicked(MouseEvent e) {
			frm_menu fr = new  frm_menu();
			fr.setVisible(true);
		}
	});
	btnNewButton.addActionListener(new ActionListener() {
		public void actionPerformed(ActionEvent e) {
		}
	});
	btnNewButton.setBounds(474, 312, 64, 23);
	contentPane.add(btnNewButton);

	JButton button_1 = new JButton("");
```

```
            button_1.setIcon(new ImageIcon("C:\\Users\\Andres
Valencia\\eclipse-workspace\\tesis\\img\\sound-frecuency.png"));
            button_1.setBounds(326, 96, 95, 78);
            contentPane.add(button_1);
            JLabel lblCaptacin = new JLabel("Captaci\u00F3n");
            lblCaptacin.setForeground(Color.WHITE);
            lblCaptacin.setFont(new Font("Comic Sans MS", Font.BOLD,
13));
            lblCaptacin.setBackground(Color.WHITE);
            lblCaptacin.setBounds(345, 181, 76, 21);
            contentPane.add(lblCaptacin);
      }
}
```