People's Democratic Republic of Algeria
Ministry of Higher Education and scientific Research
Hassiba Benbouali University of Chlef
Faculty of Exacte Sciences and informatics
Department of Computer Science
LMA laboratory

THESIS

SUBMITTED FOR THE DIPLOMA OF

DOCTORAT

Field : Computer Science
Speciality : Network & Security

By

Mr. Abdelouahab Nouar

CONCEPTION ET IMPLÉMENTATION D'UN PROTOCOLE SÉCURISÉ LORAWAN UTILISÉ DANS LES APPLICATIONS CRITIQUES DANS LE DOMAINE IOT

Examined on .. .. 2025, by the jury :

Mr.	ARIDJ MOHAMED	M.C.A UHBC - Chlef	Chairman
Mr.	TAHAR ABBES MOUNIR	Professor UHBC - Chlef	Supervisor
Mme.	BOUMERDASSI SELMA	Professor Paris University, French	Co-Supervisor
Mr.	BECHAR RACHID	M.C.A UHBC - Chlef	Examiner
Mr.	BRAHAMI MENOUAR	Professor ENPO - Oran	Examiner
Mr.	DOUGA YASSINE	M.C.A University Saad Dahlab Blida 1	Examiner

University year : 2025 - 2026.

People's Democratic Republic of Algeria
Ministry of Higher Education and scientific Research
Hassiba Benbouali University of Chlef
Faculty of Exacte Sciences and informatics
Department of Computer Science
LMA laboratory

THESIS

SUBMITTED FOR THE DIPLOMA OF

DOCTORAT

Field : Computer Science
Speciality : Network & Security

By

Mr. Abdelouahab Nouar

DESIGN AND IMPLEMENTATION OF A SECURE LORAWAN PROTOCOL USED IN CRITICAL APPLICATIONS IN THE IOT DOMAIN

Examined on .. .. 2025, by the jury :

Mr.	ARIDJ MOHAMED	M.C.A UHBC - Chlef	Chairman
Mr.	TAHAR ABBES MOUNIR	Professor UHBC - Chlef	Supervisor
Mme.	BOUMERDASSI SELMA	Professor Paris University, French	Co-Supervisor
Mr.	BECHAR RACHID	M.C.A UHBC - Chlef	Examiner
Mr.	BRAHAMI MENOUAR	Professor ENPO - Oran	Examiner
Mr.	DOUGA YASSINE	M.C.A University Saad Dahlab Blida 1	Examiner

University year : 2025 - 2026.

In memory of my great mother

To my father and my little family

ACKNOWLEDGEMENTS

First and foremost, I thank God for all the blessings. This work would not have been possible without the help of many interesting people who guided and inspired me, whom I would like to thank:

First, My sincere gratitude to my thesis director, who helped me most, Mr. TAHAR ABBES Mounir Professor at UHBC university, I am really thankful for his support and guidance in my research. I would like to thank him for the availability, encouragement, kindness, and patience. I would also like to thank him for proposing this subject to me, for attention and his contribution to the success of this thesis, being of great help, I was extremely impressed by his human qualities of listening and understanding throughout this work. It was a real pleasure to work with him. I extend all my gratitude to my thesis co-director, Mme. BOUMERDASSI Selma Professor at Paris University, for his advice and guidance. It would be difficult to complete this thesis without their guidance.

My sincere thanks to Mr. ARIDJ Mohamed Doctor at UHBC university for agreeing to chair this jury and all the members who kindly took part in this work.

To continue, I would like to thank all the teachers in the Computer Science Department and the administrative staff of the Hassiba Benbouali University for their efforts.

I would also like to thank all my friends for their moral support. I would therefore like to take this opportunity to thank the doctoral students: Chaib Mustapha, Bouhdjeur Ibrahim, Chenaoui Ali, Belhireche Mohamed and Djallal Srandi Mohamed for their encouragement, support and assistance.

And with a great emotion I would like to thank my wife for her patience and encouragement, and my children Mouaadh, Doua and Yassamine.

Finally, my affection goes out to my family, and in particular to my father, my sisters and brothers for their ongoing support.

Chlef, 14.10.2025.

Abstract: Of the available wireless transmission technologies, LPWANs (Low Power Wide Area Networks) are attracting increasing attention, not least because of their long radio range and low energy consumption. However, trying to minimize power consumption can sometimes compromise the resilience of data transmission in the face of environmental disturbances (interference, obstacles) and the mobility of connected objects. However, attempting to reduce power consumption can occasionally compromise the mobility of linked items and the robustness of data transmission against environmental disruptions (obstacles, interference). Additionally, each node's duration occupancy of the frequency band is limited by the long radio range (e.g. duty cycle limited to 1%).

We concentrate on LoRa/LoRaWAN technologies in this thesis. LoRaWAN may be able to accommodate a wide range of IoT applications and situations because to its numerous adjustable characteristics. It can converge to an ideal configuration to save energy due to its ADR (Adaptive Data Rate) function. More recently, LoRa and LoRaWAN have also drawn interest from applications utilizing mobile nodes. This thesis's first contribution is its presentation of how mobility affects LoRaWAN performance. To achieve this, the research consists of two (02) parts: In the first part, we present an in-depth analysis of LoRaWAN performance evaluation in a mobility context. To do this, we consider several scenarios and performance evaluation measures using the $NS-3$ simulator, based on the three mobility models most widely used in the literature, such as the Gauss Markov Mobility Model, the Random Waypoint Mobility model and the Constant Position Mobility Model. The new study presents the influence of these three models on energy consumption, PDR, network size and Radius. In order to validate the simulation results, in the second part we carried out numerous experiments with the Lora *CubeCell HTCC-AB01* in various scenarios, analyzing the *RSSI* (Received Signal Strength Indicator) level in urban and rural areas using a large number of trajectories. The maximum distance obtained in a rural area is 1310 meters of line of sight, while in an urban area, the distance is equal to 966.97 meters of line of sight. In terms of energy consumption, the results show that the GM model is 0.1 J, and for the RWP and CP it is 0.4 J, which equals 9.6 J in 24 hours, which makes the GM model four times more efficient. The GM model with Alpha = 1 performs better than the other two models, and Alpha = 0.5 performs even better in terms of PDR. At the same time, the RWP demonstrates positive results regarding delays.

In order to find the best combination for packet transmission, we provide a second contribution in this thesis to the description of the SFs allocation scheme for Lorawan. Furthermore, even though the various simulations carried out and the results obtained, we consider it more appropriate to use mathematical and logical methods to validate the *ADR* mechanism, and this was the subject of our third contribution. We used *Event-B* (the formal method) to model the protocol layers and their properties, and *Event-B* invariants to ensure protocol consistency, and we'll add more guarantees to the validity of the protocol, focusing on the formal validation of the *ADR* mechanism on the network server side.

The security aspect is studied, by presenting the physical structure of Lora packet, the Mac message types and the different LoraWan Mac commands, followed by the two activation modes of Lora End Devices including *OTAA* and *ABP* Mode. In all the literature, to our knowledge, currently the deployment of cryptography using the *NS-3* simulator has not yet achieved, this is the subject of our main and last contribution, our goal is the implementation of *AES-128* bits algorithm under the *NS-3* simulator, and to assess the performance of the LoRaWAN network in terms of energy consumption, transmission latency, packet delivery rate, and CPU utilization. The findings indicate a shift in the transmission delay from 0.25 ms to 1.7 ms for a packet of 12 bytes and 216 bytes respectively. Even though security has negative effects on network performance, the trade-off is necessary.

Keywords: LPWAN, Lora, LoraWan, ADR, Mobility Model, AES-128, CubeCell HTCC-Ab01, NS-3, Security, IoT.

Résumé : Parmi les technologies de transmission sans fil disponibles, les réseaux LPWAN (Low Power Wide Area Networks) suscitent une attention croissante, notamment en raison de leur longue portée radio et de leur faible consommation énergétique. Toutefois, les tentatives de réduction de la consommation d'énergie peuvent parfois compromettre la mobilité des éléments liés et la robustesse de la transmission des données contre les perturbations environnementales (obstacles, interférences). En outre, la durée d'occupation de la bande de fréquences par chaque nœud est limitée par la longue portée radio (par exemple, le cycle d'utilisation est limité à 1%).

Dans cette thèse, nous nous concentrons sur les technologies LoRa/LoRaWAN. LoRaWAN peut s'adapter à un large éventail d'applications et de situations IoT grâce à ses nombreuses caractéristiques ajustables. Peut converger vers une configuration idéale pour économiser de l'énergie grâce à sa fonction ADR (Adaptive Data Rate). Plus récemment, LoRa et LoRaWAN ont également suscité l'intérêt des applications utilisant des nœuds mobiles. La première contribution de cette thèse est la présentation de la façon dont la mobilité affecte la performance de reseau LoRaWAN, pour y parvenir, la recherche se compose de deux (02) parties : Dans la première partie, nous présentons une analyse approfondie de l'évaluation des performances de LoRaWAN dans un contexte de mobilité. Pour ce faire, nous considérons plusieurs scénarios et mesures d'évaluation des performances à l'aide du simulateur $NS-3$, basés sur les trois modèles de mobilité les plus utilisés dans la littérature, tels que le modèle de mobilité de Gauss Markov, le modèle de mobilité de points de cheminement aléatoires et le modèle de mobilité de position constante. La nouvelle étude présente l'influence de ces trois modèles sur la consommation d'énergie, le PDR, la taille du réseau et la distance du GW. Afin de valider les résultats de la simulation, dans la deuxième partie, nous avons réalisé de nombreuses expériences avec le module Lora *CubeCell HTCC-AB01* dans divers scénarios, en analysant le niveau de *RSSI* (Received Signal Strength Indicator) dans les zones urbaines et rurales en utilisant un grand nombre de trajectoires. La distance maximale obtenue dans une zone rurale est de 1310 mètres de vol d'oiseau , tandis que dans une zone urbaine, la distance égale à 966,97 mètres. En termes de consommation d'énergie, les résultats montrent que le modèle GM est de 0,1 J, et pour le RWP et le CP, il est de 0,4 J, soit 9,6 J en 24 heures, ce qui rend le modèle GM quatre fois plus efficace. Le modèle GM avec Alpha = 1 est plus performant que les deux autres modèles, et Alpha = 0,5 est encore plus performant en termes de PDR. Parallèlement, le RWP présente des résultats positifs concernant le delais.

Afin de trouver la meilleure combinaison pour la transmission de paquets, nous avons apporté une deuxième contribution dans cette thèse à la description du schéma d'allocation des SFs pour Lorawan. De plus, même au vu des différentes simulations réalisées et des résultats obtenus, nous considérons qu'il est plus approprié d'utiliser des méthodes mathématiques et logiques pour valider le mécanisme ADR, et cela a fait l'objet de notre troisième contribution. Nous avons utilisé *Event-B* (la méthode formelle) pour modéliser les couches de protocole et leurs propriétés, et les invariants *Event-B* pour assurer la cohérence du protocole, et nous ajouterons plus de garanties à la validité du protocole, nous concentrant sur la validation formelle du mécanisme ADR côté serveur.

L'aspect sécurité est étudié en présentant la structure physique des paquets Lora, les types de messages Mac et les différentes commandes Mac LoraWan, puis les deux modes d'activation des terminaux Lora, notamment les modes $OTAA$ et ABP. À notre connaissance, le déploiement de la cryptographie à l'aide du simulateur *NS-3* n'a pas encore été réalisé dans la littérature. C'est l'objet de notre principale et dernière contribution. Notre objectif est d'implémenter l'algorithme AES-128 bits sous le simulateur *NS-3* et d'évaluer les performances du réseau LoRaWAN en termes de consommation d'énergie, latence de transmission, de débit de livraison des paquets et d'utilisation du processeur. Les résultats indiquent un décalage du délai de transmission de 0,25 ms à 1,7 ms pour un paquet de 12 et 216 octets respectivement. Même si la sécurité a des effets négatifs sur les performances du réseau, un compromis est nécessaire.

Mots clés: LPWAN, Lora, LoraWan, ADR, Modèle de mobilité, AES-128, CubeCell HTCC-Ab01, NS-3, Sécurité, Iot.

Contents

List of Figures

List of Tables

Acronyms

ABP Activation By Personalization. 87, 94, 104

ADR Adaptive Data Rate. 66, 74

AES Advanced Encryption Standard. 90, 92

AES-CMAC AES Cipher-based Message Authentication Code. 110

AppEUI Application Identifier. 91

AppKey Application Key. 99

AppSKey Application Session Key. 90, 110

B-ADR Blind ADR. 76

BPSK Binary Phase Shift Keying. 6

BW Bandwidth. 13, 16

CF Carrier Frequency. 16

CR Coding Rate. 16, 94

CRC Cyclic Redundancy Check. 94

CSS Chirp Spread Spectrum. 8, 12, 14, 16

DevEUI Device Identifier. 91

DL Downlink. 76

ED End Device. 6

ETSI European Telecommunications Standards Institute. 13

FDMA Frequency Division Multiple Access. 7

GW Gateway. 15

IoT Internet of Things. 6, 12

ISM Industrial, Scientific, and Medical. 10, 12

LoRa Long Range. 8, 12, 16

LoRaWAN Long Range Radio Wide Area Network. 8, 90

LPWAN Low Power Wide Area Network. 6, 12

M2M Machine to Machine. 6, 12

MIC Message Integrity Code. x, 88, 110

NIST National Institute of Standards and Technology. 90

NS Network Server. 75

NS-3 Network Simulator 3. 47, 52

NwkKey Network Key. 102

NwkSKey Network Session key. 90

OTAA Over The Air Activation. 87, 94

PDR Packet Delivery Ratio. 67

QoS Quality of Service. 8

QPSK Quadrature Phase Shift Keying. 7

RSSI Received Signal Strength Indicator. 11

SF Spreading Factor. 13, 14

SNR Signal to Noise Ratio. 16

SVM Support Vector Machine. 67

ToA Time on Air. 16, 73

TP Transmission Power. 16

UL Uplink. 72, 75

UNB Ultra Narrow Band. 6

WSN Wireless Sensor Network. 6, 12

Introduction

In this chapter, we introduce the context of our research and the objectives we have set ourselves. We set out the questions that gave rise to the research topic and the research problem addressed. Finally, we conclude this chapter with the structure of the manuscript.

.1 Context and Motivation

The market for the Internet of Things (IoT) is becoming more and more appealing thanks to the multiplicity of applications and the suitable technologies available nowadays. The number of IoT-connected devices has grown from 8.6 billion in 2019 to 11.3 billion in 2021 and is projected to reach 30 billion in 2030 [1]. Nowadays, this technology is stimulating the development of new applications and has become essential in developments such as smart buildings, smart transport systems, smart cities, smart agriculture and so on. These vast areas of application have attracted both the market and the research and innovation community to the " Smart World ", where the idea is to connect people, objects and animals with the services they use every day.

The exponential number of connected objects and the development of intelligent environments mean that a large number of heterogeneous devices need to be interconnected and the scalability of the network needs to be managed, while guaranteeing satisfactory quality of service (QoS). However, to ensure this QoS and to respond to the different criteria of the IoT, the traditional architecture of the Internet is considered rigid and does not lend itself to these new QoS requirements. Optimal and satisfactory end-to-end QoS in the IoT is denied by long range, low energy consumption and cost-effectiveness. Gateways implement several communication protocols that have a direct impact on improving or degrading the QoS of the IoT network. The short-range radio technologies widely used in smart homes, for example, are not suited to scenarios requiring long-distance transmission. On the other hand, solutions based on cellular communications can provide greater coverage, while consuming far too much energy. As a result, the requirements of IoT applications have led to the emergence of a new wireless communication technology: the Low Power Wide Area Network (LPWAN).

This new communication technology is becoming increasingly popular in industrial and research circles due to its low-power communication characteristics, extensive coverage, low cost and large-scale deployment, which meet IoT criteria.

In this market, there are already several LPWAN solutions, such as NB-IoT, which follow 3GPP standards and operate on licensed bands. Additionally, unlicensed solutions, out of 3GPP standards, which operate on the industrial, scientific, and medical (ISM) radiobands are available, such as Sig-Fox and LoRaWAN.

Among the various LPWAN technologies, we are interested in LoRa technology, which is becoming increasingly widespread since with a powerful and robust physical layer, named LoRa, and due to its accessibility based on open source code dedicated to the research field. The LoRa network enables communication over long distances, is highly energy-efficient (battery life of over 10 years [1]) and inexpensive. However, the proliferation of these connected objects that handle sensible data puts the lives of their users at peril. This information could be used by unauthorised individuals seeking to make illicit profits from it. This is why the security and confidentiality of user data represent major challenges for the IoT. What's more, the use of the Internet of Things introduces some challenges for the implementation of traditional security methods. On the IoT side, the scarcity of resources in terms of energy, memory and computing capacity that characterises connected objects limits the deployment of traditional security solutions. A first requirement would be to use authentication mechanisms as a first barrier to

the threats faced by IoT communications to guarantee the legitimacy of the communicating entities. These authentication mechanisms must be based on lighter encryption algorithms that are adapted to the resources of the connected objects used.

These promising aspects have led to recent experimental studies on the performance of LoRa technology in outdoor and indoor environments. However, a number of questions remain to be answered:

- Is this technology capable of satisfying all the requirements of the IoT at the same time ?
- Can we consider deploying this technology in different environments, including mobility for example ?
- What are the parameters influencing QoS and how can we optimise their configuration for better QoS with very low complexity ?
- Can this technology guarantee the security and confidentiality of user data ?
- Can this technology guarantee the authentification and security of network entities ?

Answering these questions is the objective of our thesis.

.2 Aim

In order to face up to the problems and challenges mentioned above, we have set ourselves the following objectives:

— Objective A : Establish a state of the art on the Internet of Things and its various technologies, usage domains and the challenges to be met in order to support its development.

— Objective B : (1) Study mobility models and their behaviour focusing on the following three models: Gauss Markov Mobility Model , RandomWayPoint Mobility Model and Constant Position Model . (2) Present the results obtained by simulation using the *NS*3 simulator relating to the impact of mobility on lorawan performance (3) present the experimental results using the *CubeCell HTCC - AB01* module in an urban and rural environment.

— Objective C : Present a state of the art on spread factor and to study its influence on lorawan peformances, the results obtained by simulation. (2) Investigate the *ADR* (Adaptative Data Rate) mechanism and proceed with formal validation (ADR server-side) using Event-B.

— Objective D : (1) Conduct an in-depth study of the security mechanisms used (2) Present the *AES-128* bits cryptography standard and describe the security issues and solutions proposed in the literature. (3) Deploy AES-128 cryptography algorithm under NS-3.

.3 Contributions & publications

This section summarizes the publications, conferences, and collaborations based on work that I have done during my Ph.D. candidature.

➤ **Journal Papers**

• **NOUAR Abdelouahab**, M. Tahar Abbes, S. Boumerdassi, and M. Chaib. Experimental results of using AES-128 in LoRaWAN. Scientific and Technical Journal of Information Technologies, Mechanics and Optics, vol. 25, no. 5, 2025, doi: 10.17586/2226-1494-2025-25-5.

• **NOUAR Abdelouahab**, TAHAR ABBES Mounir , BOUMERDASSI Selma, et al. Impact of Mobility Model on LoRaWan Performance. Journal of Communications, 2024, vol. 19, no 1.

• Mostefa, C., Mounir, T. A., Abdelmadjid, A. M., & **Nouar Abdelouahab**. (2024). FT-CSMA: A Fine-Tuned CSMA Protocol for LoRa-Based Networks. Journal of Communications, 19.2.

➤ **International Conference Papers**

• C. Mostefa, **Nouar Abdelouahab**, T. A. Mounir, S. Boumerdassi, S. Femmam and Z. A. Amel, "Formal Validation of ADR Protocol in LoraWan Network Using Event-b," 2023 7th International Conference on Computer, Software and Modeling (ICCSM), Paris, France, 2023, pp. 11-15, doi: 10.1109/ICCSM60247.2023.00011.

• **Nouar Abdelouahab**, TAHAR ABBES Mounir , Chaib Mostefa, "Usage of AES-192 encryption in LoRaWAN network under NS-3" 2025, The first international Conference on Artificial Intelligence, Smart Technologies and Communications (AISTC'25) Chlef, Algeria, 2025.

➤ **National Conference Papers**

• **Nouar Abdelouahab**, T. A. Mounir, C. Mostefa "Spreading Factor Assignment Scheme on LoRaWAN Network" First National Conference on " Artificial Intelligence, Smart Technologies and Communication", Chlef, Algeria, November 08th & 09th, 2023.

.4 Thesis Structure

The rest of this thesis manuscript is organized into four chapters.

• **Chapter 1:** Firstly, an overview of the technological solutions for the IoT is presented, focusing on low-power wide-area networks (LPWAN). The basic characteristics and operation of the various LPWAN standards are discussed, along with their fields of use. Next, the LoRa standard and its MAC protocol (LoRaWAN), on which this thesis is based, are discussed in detail. This chapter responds to **Objective A**.

• **Chapter 2:** first presents the mobility models followed by the results showing the impact of mobility on lorawan performance obtained by the $NS3$ simulator. Then we present the experimental results using *CubeCell HTCC-AB01* in a rural and urban environment. This chapter responds to **Objective B**.

• **Chapter 3:** In this chapter, we present a state of the art of spread factor and how to obtain the best combinations to optimize the global Qos of the network, subsequently an invistigation on the ADR mechanism, at the end of the chapter we proceed to a formal validation of the ADR server side using the *Event-B* . This chapter responds **Objective C**.

• **Chapter 4:** This chapter is devoted to security in Lorawan. To achieve **Objective D**, we study the two End Device activation modes, $OTAA$ and ABP for all versions of Lorawan, give an overview of the AES standard used, then we describe our contribution, which is the implementation of $AES-128$ bits encryption under the $NS-3$ simulator. The security open issus and challenges, and the security needs that must be guaranteed to counter them.

IoT & LoRaWAN Technologies I

Contents

With the emergence of a large variety of applications aimed at making people's lives more comfortable and the environment more intelligent, different communication technologies, mainly radio, are being used to connect thousands of objects to the Internet. The aim of this chapter is to provide an overview of new technological solutions for Low Power Wide Area Network (LPWAN) by presenting LoRa, Sigfox and NB-IoT, followed by a comparative study and application examples, answering the question of which technology fits best. After describing the LoRa physical layer and its different parameters, as well as the *CSS* modulation method, the LoRaWAN MAC layer is describing, which is the topic of our research, and explains why this technological choice was made. Finally, we discuss the open issues in LoRa networks.

I.1 Introduction

There has been a noticeable increase in the number of devices we use on a daily basis, as well as the desire to connect to the Internet. With time, this expansion leads to the development of the so-called Internet of things IoT paradigm. Health, transportation, smart homes and cities, agriculture, and a number of other fields are among the many application areas for IoT [2]. In order to access actionable data, the Internet of Things (IoT) combines a number of devices, networks, software, and sensors. It is not a single technology.

An extensive array of technology facilitates Internet of Things connectivity. The most popular ones are cellular networks (like 2G, 3G, and 4G), wireless personal area networks (like Bluetooth), wireless local area networks (like Wi-Fi), and Low Power Wide Area Network (LPWAN) (like LoRa and SIGFOX). To determine each technical solution's specific area of application, these many technologies are arranged in Fig.I.1 based on their range.

LPWANs are wireless communication technologies ideal for Internet of Things and M2M applications because they allow for long-distance communications with low power consumption [3], a low-cost interface, and relatively low bit rates. LPWAN solutions typically deploy a large number of End Devices (ED), send limited message sizes and tolerate relatively long end-to-end delays.

I.2 LPWAN Technologies

An emerging wireless network architecture called low-power wide-area networking (LPWAN) seeks to give Internet of Things (IoT) devices long-range connectivity [2]. A LPWAN's architecture consists of networks, application servers, base stations, and end devices. Compared to other wireless network systems, end devices are made to be able to communicate with base stations across larger distances. The energy consumption of nodes in these kinds of networks has a significant influence on how well they operate.

Being a WSN-type of network, the generic LPWAN has some specific characteristics defined by association, namely the type of communication and sensor specifications. A major difference compared to other WSNs consists of network topology. LPWAN is based on star topology, each sensor communicating with the associated base-station, which, in turn, forwards the information into the network, towards the network server. The LPWAN concept for the information flow eases the sensor of the burden of information processing, helping in leveraging power resources in remote areas covered by them. The goal of low-power wide area networks, or LPWANs, is to enable end devices with limited power (such as those that rely on energy harvesting or batteries) to function for years at low data rates, connecting to gateways located several kilometers away. The Internet of Things will become much more connected thanks to LPWAN technology, which will also make it possible for IoT items to deeply penetrate metropolitan areas. Comparisons of different wireless technologies with respect to radio power consumption and communication ranges are shown in Fig.I.1. The image illustrates how LPWANs, such as LoRaWAN [3], Sigfox [4], and NB-IoT [5], constitute a significant pole in the radio power consumption versus communication range spectrum.

I.2.1 Sigfox.

Sigfox is the first open-source technological system of its sort. Sigfox, which uses Ultra Narrow Band (UNB) frequency and deploys its proprietary base stations in different countries in the unlicensed sub-GHz ISM bands (e.g., 868 MHz in Europe, 915 MHz in North America, and 433 MHz in Asia), was designed to focus on an even longer range and lower data rate compared to LoRaWAN. Sigfox can be used to implement star topology. Using the Binary Phase Shift Keying (BPSK) modulation in an ultra narrow band of 100 Hz at a maximum data rate of 100 bps, the end-devices establish a connection with these base stations [6]. By

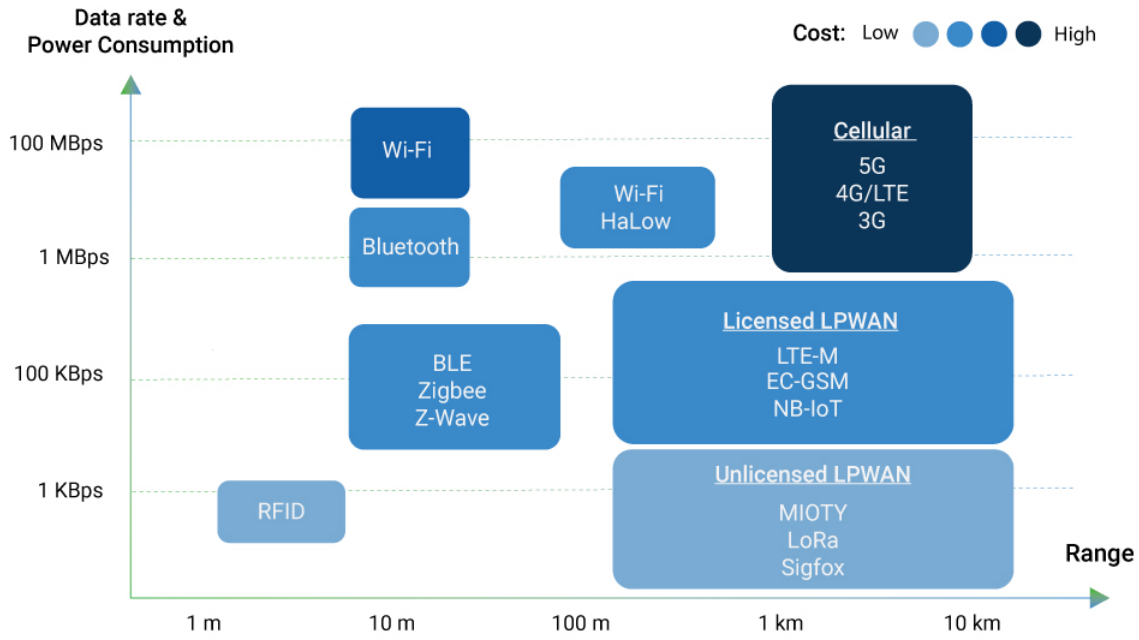


Fig. I.1 – *Positioning of LPWAN concerning Power consumption & data rate versus communication range.*

employing ultra narrow band in sub-GHz spectrum, Sigfox efficiently uses the frequency band and has very low noise levels, leading to very low power consumption, high receiver sensitivity, and low cost antenna design.

Sigfox was originally limited to uplink transmission, but it was eventually extended to allow bidirectional communication. Only after an uplink broadcast does the downlink transmission take place. There is a 140 messages daily cap on the amount of messages sent over the uplink. Each uplink message can have a maximum payload length of 12 bytes. Nevertheless, there is a daily cap of four messages per downlink user. Each downlink message can have a maximum payload length of 8 bytes. If a connected device moves, there is no roaming because it is using the same network provider [7].

I.2.2 NB-IoT.

The objective of the IoT communication protocol known as NB-IoT is to improve the features required for IoT networks while at the same time minimizing the capabilities of LTE. While NB-IoT is compatible with low-power IoT devices, it is a great option for IoT applications. A narrowband LPWAN called NB-IoT is capable of coexisting in licensed frequency bands with either GSM or LTE. 200 KHz is the frequency bandwidth that NB-IoT uses, which is equivalent to one resource block in GSM and LTE transmission [8].

NB-IoT communication protocol is based on LTE protocol (i.e. NB-IoT reuses various principles and building blocks of LTE physical and higher protocol layers). In fact, NB-IoT reduces LTE protocol functionalities to the minimum and enhances them as needed for IoT applications. NB-IoT allows connectivity of more than 100K devices per cell and it could be increased by exploiting multiple NB-IoT carriers. The bits are modulated using Orthogonal FDMA (OFDMA) in the downlink, Frequency Division Multiple Access (FDMA) in the uplink, and Quadrature Phase Shift Keying (QPSK) modulation scheme [8]. The highest throughput rate is 200 kbps for downlink and 20 kbps for uplink. Each message has a payload size of 1600 bytes, and by

sending 200 bytes a day on average, the battery may last 10 years. In actuality, Release 15 of 3GPP continues to develop NB-IoT. The current 3GPP plan calls for the extension of NB-IoT to incorporate multicast services.

I.2.3 LoRa / LoRaWAN.

Growing industry enthusiasm for LPWANs has been inspired by SIgFox's success. The LoRa (Long Range) technology that supports Long Range Radio Wide Area Network (LoRaWAN) was initially created by the Grenoble, France-based startup Cycleo in 2009 and acquired by the Semtech (USA) group three years later in 2012. More than 500 firms are members of The LoRa Alliance, a non-profit organization that has been standardizing LoRaWAN since 2015.

LoRa, like SigFox, operates on an unlicensed ISM frequency range [7], use 868 MHz in Europe and 915 MHz in North America as frequencies. It is crucial to make clear that LoRaWAN represents the protocol layer and LoRa represents the physical radio frequency layer. In addition to utilizing Chirp Spread Spectrum (CSS) for spread spectrum modulation and bidirectional data transmission, LoRaWAN also incorporates end-to-end encryption with network and application keys to preserve data integrity and guarantee security. LoRaWAN can cover up to 20 km in rural regions and up to 5 km in urban areas, with a maximum throughput of 22 kbits/s. It's noteworthy to note that LoRaWan allows for the adaptation of both data rate and range via the use of a spreading factor. Increasing this factor increases the transmission range but comes at the cost of data rate and vice versa.

I.2.4 Comparaison Regarding IoT Factors.

The main goal of an LPWAN network is to offer long-range, low-power connection. Although certain LPWAN systems offer lower costs and better power usage, others offer higher Quality of Service (QoS) [6, 9]. When constructing an LPWA network for Internet of Things applications, numerous considerations need to be taken into account. These variables frequently include cost, payload length, range of coverage, battery life and latency, and quality of service (QoS). The main distinctions and similarities between the LPWAN networks previously described are displayed in Table I.1.

• **Power consumption and battery life:** Numerous factors influence power usage, which in turn affects how long a device's battery lasts. The access method has the biggest impact: Sigfox and LoRaWAN class A use basic Aloha, where nodes spend the majority of their time in sleep mode, which reduces power consumption and extends battery life. NB-IoT, on the other hand, uses more intricate and synchronized access techniques, which demand more power.

• **Quality of Service(QoS):** Asynchronous communication based on the ALOHA protocol and license-free sub-GHz channels are used by Sigfox and LoRaWAN. They don't offer quality of service, but they can effectively bounce interference and fading/multi-path. At the expense of cost, NB-IoT uses synchronous LTE protocols and licensed spectrum, which are best for QoS. For applications that require guaranteed quality of service, NB-IoT is recommended because to the tradeoff between QoS and cost; for applications without this requirement, LoRaWAN or Sigfox should be used.

• **Scalability & Payload size:** One of the main characteristics of NB-IoT, LoRaWAN, and Sigfox is their ability to support thousands of end devices. High scalability is a feature offered by these cellular LPWAN networks. When compared to Sigfox and LoRaWAN, NB-IoT has the advantage of extremely high scalability. Compared to Sigfox and LoRaWAN, which only support 50K devices per cell, NB-IoT enables connectivity for over 100,000 devices per base station. In contrast, LoRaWAN packet lengths range from 0 to 259 bytes, contingent upon the spreading factor, whereas NB-IoT allows for data transmission up to 1600 bytes. On the other

hand, Sigfox suggests a limited payload length of 12 bytes, which would restrict its applicability to a variety of IoT applications that require the transmission of larger amounts of data.

Table I.1 – *Overview of LPWANs technologies: Sigfox, LoRaWAN, and NB-IoT [10].*

	LoRa/LoRaWAN	**Sigfox**	**NB-IoT**
Frequency (frequency bands)	863–870 MHz (unlicensed)	863–870 MHz (unlicensed)	GSM 900 MHz, LTE 800 MHz (licensed)
Modulation	CSS	GFSK/DBPSK	QPSK
Bandwidth	0.3–11 kbps <50 kbps (FSK)	UL: 100 bps DL: 600 bps	20-250 bps
Max messages per day	Unlimited	140 Up, 4 Down	Unlimited
Max payload length	243 bytes	12 bytes UL, 8 bytes DL	1600 bytes
Range	6–30 km	10–40 km	2–35 km
Bidirectional	Yes / Half-Duplex	Limited / Half-Duplex	Yes / Half-Duplex
Interference Immunity	Very High	Very High	Low
Confidentiality	AES-128	Default: None Optional: AES-128	LTE
Encryption	Yes	No	Yes
Link budget (db)	154	159	151
Battery life	10 years	10–20 years	10 years

• **Network coverage:** With the maximum Link Budget (168 dB and 164 dB), NB-IoT can cover distances of 1–5 km in cities and 10–50 km in rural areas. Moreover, NB-IoT cannot be used in areas lacking LTE coverage, such as rural areas, because it can only be used within LTE infrastructure. In urban areas, LoRa and Sigfox offer 150 dB and 156 dB of Link Budget, respectively, covering 2-10 km and 15-50 km in rural areas. [6].

• **Cost:** The price of an LPWAN network comprises the following: base station deployment costs, spectrum costs (if licensed), and end device costs. Because they make use of unlicensed spectrum and less expensive gateways and end devices, LoRa and Sigfox have the benefit of being less expensive.

Compared to Sigfox and LoRa, NB-IoT is far more expensive because it requires a licensed frequency spectrum for operation. As the most expensive of the three LPWAN protocols, NB-IoT is also more expensive than LoRa and Sigfox due to the higher cost of its end devices.

• **Deployment model:** June 2016 saw the introduction of the NB-IoT specs, therefore more time will pass before its network is set up. However, the ecosystems for Sigfox and LoRa have reached maturity and are currently being commercialized in a number of nations and cities. Sigfox is currently only adopted in 31 countries, whereas LoRa is currently deployed in 42 countries thanks to its advantage [9, 11]. However, the global implementation of Sigfox and LoRa is still ongoing. Furthermore, the flexibility of the LoRa ecosystem is a major benefit. LoRa provides both public network operation through base stations and local network deployment, i.e., LAN utilizing LoRa gateway, in contrast to Sigfox and NB-IoT. In the sphere of industry, a hybrid operating model could be used to deploy a local LoRa network in factory areas and uses the public LoRa network to cover the outside areas.

• **Flexibility:** LoRaWAN is an open-source standard that encourages researchers and developers to construct their networks and contribute to the evaluation and improvement of LoRaWAN, despite its patent-based foundation. Knight has also created an software-defined radio (SDR) solution that makes use of the LoRa physical layer being reversed, as well as increased access for open source developers to investigate and refine the technologies. Conversely, certain technologies, such as Sigfox, are exclusively private, and others, like NB-IoT, are cellular in nature, requiring users to subscribe to and test the network provider.

We choose LoRaWAN specifically for our research because it uses the license-free ISM band, is an open data link standard, is an unmanaged network, which eliminates the need for infrastructure, and has been extensively used for various kinds of applications (see Fig.I.2).

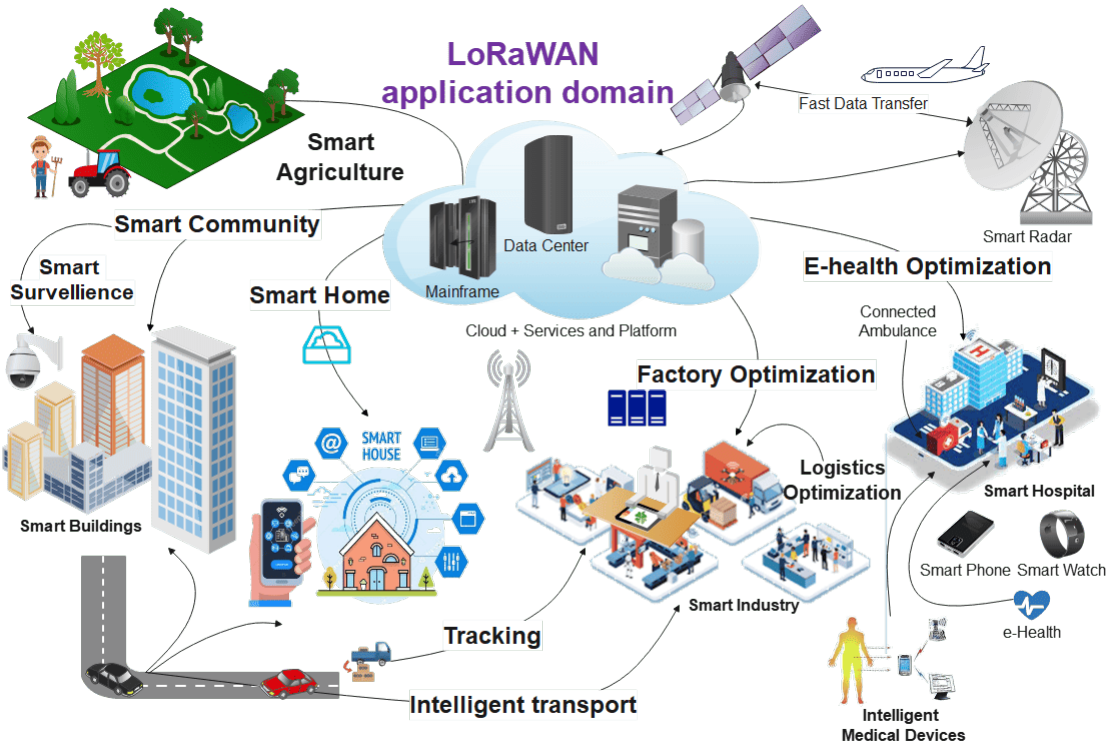


Fig. I.2 – *Top applications of LoRaWAN technology*

I.2.5 Application examples:

Which technology is best for a given application will depend on an array of IoT characteristics and the technological distinctions between LoRaWAN, Sigfox, and NB-IoT. One technology is not suitable for every Internet of Things application, as this study has already demonstrated. Many application use cases are covered in this part along with a synopsis of the most appropriate technology.

• **Electric metering:** Companies usually need high data rates, minimal latency, and frequent communication in this application field. Since electric meters provide a constant power source, they often don't require low energy usage or lengthy battery lifetimes. In addition, businesses require real-time grid monitoring in order to make quick choices regarding load, outages, and interruptions. Since Sigfox cannot tolerate low latency, it is therefore unsuitable for this application [6]. On the contrary, electric meters can be set up using LoRaWAN Class-C to assure very low latency. But because this application needs frequent communication at a high data rate, NB-IoT is a better fit. Furthermore, in densely populated places, electric meters are usually stationary. Then, cellular operators may easily guarantee NB-IoT coverage (LTE).

• **Smart farming:** Sensor devices must have a long battery life in the agricultural industry. Alkalinity, temperature, and humidity sensors have the potential to greatly lower water usage while increasing yield. Since there hasn't been a significant change in the environment, the devices update sensed data a few times per hour. For this reason, LoRa and Sigfox are perfect for this application. Moreover, NB-IoT is not the answer for agriculture in the near future because many farms do not currently have LTE cellular service.

• **Manufacturing automation:** Industrial production lines are kept running smoothly by real-time machinery monitoring, which also enables remote control for increased productivity.

There are many kinds of sensors and communication needs in factory automation. For certain applications, NB-IoT is a better option than Sigfox and LoRaWAN since it offers high quality of service and frequent communication. Other uses, including asset tracking and status monitoring, call for inexpensive sensors with a long battery life; in these situations, Sigfox and LoRaWAN work well. Owing to the diversity of needs, hybrid solutions may also be employed.

• **Intelligent Logistics and Transportation** Two crucial pillars that will support the expected growth of the Internet of Things in the upcoming years are logistics and transportation. When delivering data to cloud servers, intelligent transportation systems may experience increased communication overhead, high bandwidth usage, and considerable reaction delays due to the massive amount of data generated by sensors installed on automobiles or roadside devices. Transportation applications have employed the cellular LPWAN designs of NB-IoT to address the aforementioned difficulties. Furthermore, backhaul infrastructure for applications involving intelligent transportation systems can be provided by NB-IoT. These applications are ideally suited for NB-IoT due to their diversity, range, QoS, and low latency.

• **Smart City:** Smart City applications are developing large-scale IoT applications including automatic lighting, intelligent parking, and smart rubbish collection. In this type of situation, LoRaWAN and Sigfox are suitable options since they can cover wide areas and a significant number of user devices at the expense of higher jitter, latency, and collision rates.

• **Smart building:** In order to prevent damages and promptly respond to requests without the need for a human building monitor, property managers are alerted by sensors for temperature, humidity, security, water flow, and electric plugs. Additionally, there is potential for more effective building upkeep and utilization. Low cost and extended battery life are requirements for these sensors. For this class of applications, Sigfox and LoRa are a better fit because they don't need frequent communication or quality of service [6].

• **Retail point of sale terminals:** Since sale-point systems manage numerous communications, they need to be provided with guaranteed quality of service. Because these systems have a constant source of electricity, the lifespan of the batteries is unaffected. Low latency is also a must; that is, a store's ability to complete a certain number of transactions is restricted by high latency times. NB-IoT is hence a better fit for this use case.

• **Smart Environment:** Innovative IoT-based ecosystems hold data on water quality, air pollution levels, temperature reduction, landslide and forest fire prevention, animal tracking, snow level monitoring, and earthquake early warning. Long-lasting batteries are necessary for this kind of project, which also considers coverage and range. But they also need a lot of bandwidth, a good quality of service, and effective interference avoidance. Furthermore, the projects are frequently completed by extensive endeavors that can justify greater costs. NB-IoT is therefore more suitable for this kind of application. Nevertheless, LoRaWAN implementation can help with smart water grid management. Long-distance communication is another application for LoRa technology. **Nouar** et al.[12] conducted an experimental study in Algeria to evaluate the LoRa communication range and Received Signal Strength Indicator (RSSI) in urban and natural settings. The results showed that LoRa is robust and suitable in dense urban environments, with a possible 966.97 m distance connection.

• **Energy Management:** A few key elements include network control, load adjusting, remote observing and estimating, transformer wellbeing checking, and observation of wind farms and sunlight-based power establishments in order to create an industrial-level smart grid energy metering environment based on IoT-technology. For high-level smart grid and energy metering, long range, low power, strong quality of service, and outstanding readability are necessary. Thus, in this case, NB-IoT is more appropriate for use. Such devices include intelligent hazard alert systems and energy bill meters. Industrial energy systems have consequently grown to play a significant role in the Internet of Things.

• **Wearables and Health:** It is easy to monitor and manage a patient's health-related parameters, manage wearable medical technology, patient surveillance, telemedicine, fall detection, athlete care, track chronic illnesses, and monitor populations of mosquitoes and other simi-

lar insects using various IoT communication protocols. Since low latency, diversity, range, and quality of service are required by most systems, NB-IoT is a perfect fit for these applications. Applications such as wearables for healthcare, patient monitoring, indoor remote health, and wellbeing monitoring can be run on shorter-range IoT protocols like Sigfox and LoRa.

• **Asset Tracking and Monitoring:** Applications for IoT-based asset tracking and monitoring are becoming more popular in the sector and are especially sought after in the logistics sector. Industrial bases and leased assets require tracking and monitoring that is both scalable and effective. There are many different tracking alternatives that may be used for anything from tracking cars and bicycles to tracking company assets like packages, service goods, and containers. The asset's exact location at the nearest gateway is ascertained using the differential time of arrival method in LoRaWAN geolocation tracking [13]. LoRaWAN and Sigfox are good choices since they can support large numbers of users and a large coverage area, but at the cost of increased jitter, latency, and collision rates.

To determine which IoT protocol is appropriate for a given set of usage criteria, Table I.2 provides an overview of the various protocols' applicability from an application perspective.

Table I.2 – *Suitable IoT communications protocol based on application and physical features.*

Use Cases	Features	Protocol
Smart City	Large coverage area, latency, cost	Sigfox, LoRaWAN
Intelligent Logistics and Transportation	Low latency, high QoS, large coverage	NB-IoT
Smart Farming	Large coverage area, latency, cost, low power	Sigfox, LoRaWAN
Smart building	Short range, lower latency, cost, low power	Sigfox, LoRaWAN
Terminals for Retail Sales	Low latency, high QoS, large coverage	NB-IoT
Smart Environment	Low latency, high QoS, large coverage	NB-IoT
Smart Metering, Energy	Long range, low power, robust QoS, and readability	NB-IoT LoRaWAN Class C
Manufacturing automation	Long range, low power, robust QoS, readability and cost	Sigfox, LoRaWAN and NB-IoT
Wearables and Health	Long range, low power, Robust QoS, readability and cost	Sigfox, LoRaWAN and NB-IoT
Asset Tracking and Monitoring	Low power, cost and readability	Sigfox, LoRaWAN

I.3 Lora Technology.

Aiming for Wireless Sensor Network (WSN) in a world where Machine-to-Machine (M2M) applications are becoming commonplace and the Internet-of-Things (IoT) is expanding, LoRa technology has drawn more attention from Low-Power Wide Area Network (LPWAN) vendors due to a few features that will be covered in more detail. Long Range (LoRa) is a PHY layer for a Low Power Wide Area Network as shown in Fig.I.3.

LoRa enables long-range transmissions (5 to 8 km in urban area and 15 to 30 km in rural scenarios) with low power consumption (up to 10 years battery lifetime depending on the use).

I.3.1 Lora

LoRa is a prime example of an LPWAN technology that uses inexpensive hardware to run on license-free industrial, scientific, and medical (ISM) channels. LoRa uses the Chirp Spreading Spectrum (CSS) modulation technique, which makes it noise-resistant and capable of long-range communication [14].

One of the important technologies employed by LPWAN is LoRa, via LoRaWAN. Using a spread spectrum modulation approach, it is a proprietary Physical (PHY) layer modulation created by Cycleo and bought by Semtech in 2012 [3]. It is an integrated Forward Error Correction (FEC) version of the Chirp Spread Spectrum (CSS). An ideal combination of extended range, low power consumption, and secure data transmission is provided by this technology. LoRa only addresses the communication stack's physical layer. An open standard that covers the MAC, network, and application layers is called LoRaWAN [3], and it is the communication protocol specifications developed by LoRa Alliance on top of LoRa technology. These two combined technologies provide low power, wide area communication between remote sensors and gateways connected in WSN targeting low bandwidth, low latency IoT applications.

The European Telecommunications Standards Institute (ETSI) defines the operational frequencies from the sub-Gigahertz unlicensed ISM (Industrial Scientific and Medical) band (Europe 433MHz & 868MHz, US 915MHz [15]), which are accessible at low cost for sensors with low-cost network infrastructure. These are notable benefits of LoRa associated with the LoRaWAN communication protocol, as defined by [3].

Physical layer characteristics and the surrounding environment have a significant impact on LoRa network performance considering the relationship between the kind of network and the concept behind LoRa-enabled networks, which is to expand network coverage over distant locations without access to the Internet or electricity.

The modulation, the spreading factor, the receiver sensitivity, and other issues are addressed by LoRa design principles. The Spreading Factor (SF) is the primary parameter of LoRa modulation, reflecting the interaction between the bit and the chip [14]. By using various spreading factors at minimal power consumption, CSS technology enables adaptable long-range communications. The LoRaWAN data rates can be adjusted from 0.3 kbps to 27 kbps by varying the SF from 7 to 12. In addition to increasing range, a greater SF also lengthens the time that a data packet is in the air. The energy consumption of wireless sensor nodes is critical to a LoRa network's overall operation. Consequently, one of the most important aspects of studying the impact of the data transmission process on node lifetime during the design of an ideal network is the energy efficiency parameters.

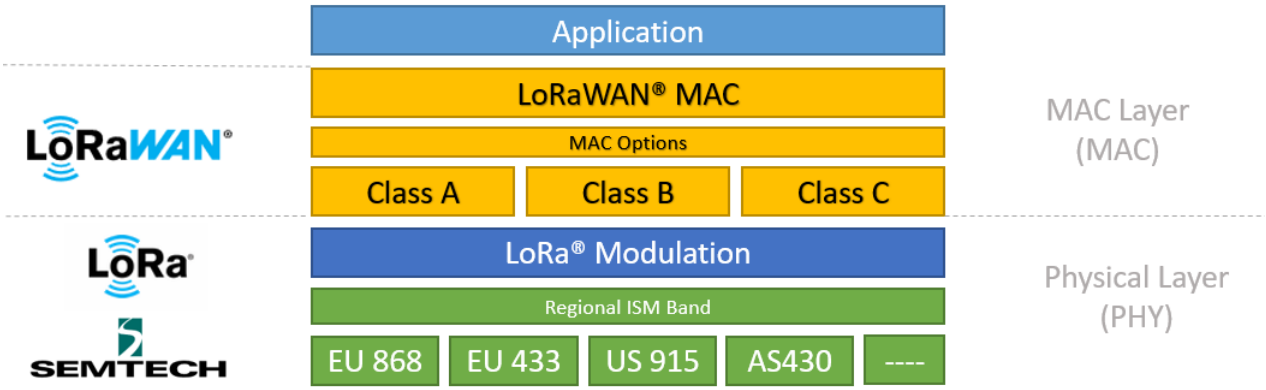


Fig. I.3 – *Lora Technology stack.*

I.3.2 The LoRa Modulation

Utilizing Spread Spectrum [3] for data transmission, LoRa modulation is a unique PHY layer technology. However, it will employ "*Chirps*" instead of "*codes*" , which is why it is termed Chirp Spread Spectrum modulation. Having many transmissions in the same channel at the same time is always the goal. The effect on the spectrum is also constant: it disperses the spectrum across the chosen Bandwidth (BW).

I.3.3 LoRa's Chirp Spread Spectrum Implementation

The concept underlying *CSS* is that information can be "*spread*" over a larger spectrum than it would typically need to occupy by using a sinusoidal signal known as a "*chirp*", which has a set duration and a frequency that varies linearly. The spreading factor (SF) and the chirp rate (CR), which are synonyms for bits per symbol (accepting values between 7 and 12) and symbol rate, respectively, are used to modify the transmission bit rate given the fixed bandwidth of these channels. At the expense of a reduced spectral efficiency, this uniform distribution of a symbol across a wider bandwidth offers resistance against frequency-selective noise and interference. In addition, CSS can withstand the Doppler effect and multi-path interference better than other, more traditional modulations with a few extra safety measures.

Fig.I.4 shows the basic form of the signal that is emitted by the LoRa modulation. Its symbol is utilized in radar technology, which is where the name "Chirp" originates (**Chirp**: **C**ompressed **H**igh **I**ntensity **R**adar **P**ulse); a (CSS) modulation that uses an up-chirp, or sinusoidal signal whose frequency fluctuates over time. On the other hand, the chirp is referred to as a down-chirp if its frequency drops with time [16]. A trade-off is made between data throughput and sensitivity because downlink channels have a fixed bandwidth of 500 kHz, whereas uplink channels have a fixed bandwidth of 125 kHz. Lora's unique orthogonal spreading factors allow the network to provide adaptive power and data rate optimizations for each end device, extending the battery life of linked end devices.

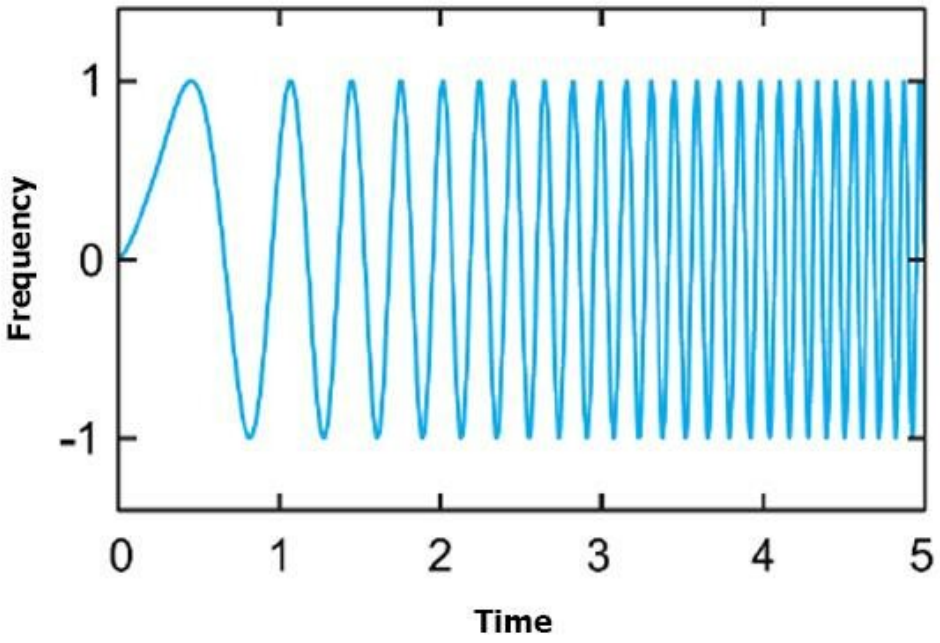


Fig. I.4 – *LoRa's Chirp Spread Spectrum (CSS) [16].*

assuming that $B = [f_0, f_1]$ is the frequency band that is available for transmission. When a chirp reaches the limit of the available band, it can be designed to wrap around from f_1 to f_0, increasing linearly in frequency from a starting frequency $f_s \in B$ to that same frequency.

It appears that a symbol in LoRa is represented by the beginning frequency of a chirp within the available bandwidth. A adjustable parameter called SF determines how many bits a symbol contains that are encoded by LoRa. This indicates that there are $M = 2^{SF}$ potential beginning frequencies for a chirp, and that a chirp employing spreading factor SF represents 2^{SF} bits using a symbol. The equation I.2 also uses the spreading factor of a transmission to calculate the duration of a symbol.

Accordingly, an increase of spreading factor of one will result in symbols that last twice as long, assuming the modulation is using a fixed bandwidth. On the other hand, a larger band-

width raises the bitrate of the modulation by increasing the rate at which chirps are broadcast. According to [17], a message becomes more resilient to noise or interference when the transmission time of a chirp (a symbol) is extended. However, while maintaining synchronization between the signal and the receiver is particularly crucial when using low data rates, this benefit might be somewhat countered by the fact that symbol errors are more likely to occur when spreading factors are higher since there are more possible symbols. An additional drawback of sending longer messages is the higher likelihood of collisions.

I.3.3.1 Sub-GHz ISM bands

Although LoRa was originally designed to operate just in the sub-GHz ISM frequency, it has lately been expanded to operate in the 2.4 Ghz ranges. At low power budgets, sub-GHz band communication is robust and dependable. In contrast to the 2.4 GHz band, obstructions and solid surfaces attenuate lower frequencies. Furthermore, compared to 2.4 GHz, which is occupied by the majority of widely used wireless technologies, including Wi-Fi, Bluetooth, ZigBee, and other household equipment, the sub-GHz band is less crowded and produces less interference.

There is no restriction on the number of devices that can operate in the sub-GHz ISM bands, making it challenging to regulate and limit their usage. Since these bands are susceptible to overuse and saturation, laws and regulations have been developed by regional and national regulatory bodies to monitor usage within them.

Any user using these bands should abide by two main rules: duty cycle and maximum transmission power. The first rule places a limit on the device's maximum transmission power (Tx). This restriction depends on the region and the subband. By the ERC (European Research Council) recommendation, Table I.3 lists the power limit per sub-band; for example, the maximum in Europe is 14 dBm (25 milliwatts), whereas the US has a 24 dBm limit on TP power. The second regulation pertains to the duty cycle restriction, which is characterized as the percentage of time that a device spends transmitting within the band.

LoRa uses ISM bands that vary depending on the area. As an illustration, in Europe, LoRa employs h1.6 in the downlink and h1.2 and h1.5 sub-bands in the uplink. Furthermore, the duty cycle varies by area and frequency and is specific to each sub-band. Duty cycles are calculated for each sub-band; in a given hour, a device may use 1% in h1.1 or h1.5, 10% in h1.7, and 1% in h1.9. The GW responds to the devices using h1.7 (cf. RX2) The duty cycle percentage for each sub-band in the 868 MHz spectrum is displayed in Table I.3 [18]. Devices can employ alternative rules such sensing the channel mechanism and Listen Before Talk. Since the percentage of restriction is per sub-band, frequency hopping can be a solution for users to improve their duty cycle. This compels radio technology to use several sub-channels within a band in order to prevent one channel from becoming saturated.

Due to its more tolerant spectrum regulations regarding radio duty cycle and ability to support higher maximum transmission power in many locations, LoRa has extended its support for the 2.4 GHz band.

Table I.3 – *ERC Recommendation for ISM bands [18].*

Sub-band	Frequency (MHz)	Transmission Power	Duty cycle
h1.1	865-868	14 dBm	1%
h1.2	863-870	14 dBm	0.1%
h1.5	868-868.6	14 dBm	1%
h1.6	868.7-869.2	14 dBm	0.1%
h1.7	869.4-869.65	27 dBm	10%
h1.8	869.7-870	7 dBm	100%
h1.9	869.7-870	14 dBm	1 %

I.3.4 Characterization of LoRa transmission parameters

The two primary components of LoRa technology are LoRa and LoRaWAN. The physical layer that Semtech presents is called LoRa. It employs forward error correction and Chirp Spread Spectrum (CSS) modulation [3] to make the signals resistant to noise and interference from the channel. The frequency of sinusoidal sounds, or chirps, sweep linearly across time. Although LoRa technology can employ a variety of bandwidth (BW) numbers, 125, 250, and 500 kHz are the most often used ones. In addition to increasing data throughput, the larger bandwidth also improves sensitivity to noise. Conversely, the SF which is the ratio of the symbol rate to the chirp rate is regarded as the most significant LoRa characteristic. The SF's value can be any integer number from 7 to 12. The transmission success is affected by the allocated SF to the end node. Higher SF increases the Signal to Noise Ratio (SNR), transmission range and Time on Air (ToA) while decreasing the data rate. According to specification [3], the transmit power in LoRa network can take value between 2 and 20 dBm, where the default value is 14 dBm. The Carrier frequency depends on region restrictions and can take value between 137 MHz and 1020 MHz. Besides, in LoRaWAN wireless communications, a receiver needs a good SNR to separate the original signal from the modulated carrier. The power of the received signal divided by the noise level is known as the SNR. It is frequently employed to assess the received signal's quality. LoRa SNR values typically range from -20 dB to +10 dB. The received signal is less distorted when the value is closer to +10 dB. In fact, transmissions that are -7.5 dB to -20 dB below the noise floor can be demodulated using LoRa.

Security mechanisms are provided by LoRa communication from the application server to the end device. Five key factors determine the performance of LoRa transmissions: Spreading Factor (SF), Bandwidth (BW), Carrier Frequency (CF), Transmission Power (TP), and Coding Rate (CR).

I.3.4.1 Carrier Frequency (CF)

represents the band's center frequency. The LoRa technology works in the unlicensed ISM band, and the CF may be programmed in increments of 61 Hz between 137 MHz and 1020 MHz. In particular, the CF changes based on geographic location; for example, in Europe, it ranges from 868 to 870 MHz, in Asia, it is 433 MHz, and in America, it is 915 MHz. While gateways normally offer eight channels, LoRa nodes often support sixteen. [19].

I.3.4.2 Spreading Factor (SF)

Chip rate divided by symbol rate equals SF. It is the total number of unprocessed bits that a symbol can have encoded in it. The signal-to-noise ratio (SNR) rises with increasing SF, which boosts sensitivity and range. Nevertheless, it causes the packet airtime to grow. According to [20], the expression 2^{SF} indicates how many chips each symbol may contain. The SF characterizes the relationship between the chip rate and the baseband data rate. The range of LoRaWAN SF values is 7 to 12, meaning that a value of 12 strengthens the communication signal by making the receiver equipment more sensitive, but the data rate drops as a result. On the other hand, when the SF is decreased, the data rate rises, but the message must have a higher TP in order for the recipient to correctly decode it. LoRa devices use a higher SF when the signal is weak, and a longer ToA is implied by utilizing a higher SF. Additionally, the distance from the the entrance affects the SF. As the end node gets farther away from the gateway, the signal gets weaker and the SF gets bigger.

SF Orthogonality: One of the primary benefits of the LoRa modulation is that, since the various spreading variables are theoretically orthogonal to one another, the receiver demodulator can distinguish between incoming concurrent broadcasts on the same frequency channel. This

avoids some of the drawbacks associated with employing a pure ALOHA channel access MAC, such as LoRaWAN, including the high likelihood of packet collision and subsequent loss of all related packets that are characteristic of pure ALOHA networks. It has been demonstrated that in actuality, the various spreading factors are only roughly orthogonal to one another. This means that packets delivered simultaneously on the same frequency but on separate spreading factors would also cause interference with each transmission. This will be developed in detail, in Section I.5.4.2, page 30.

I.3.4.3 BandWidth (BW)

The LoRa communication can operate in one of $BW \in \{125, 250, 500\}$ kHz depending on the regional parameters [21]. A LoRa modulated signal comprises of 2^{SF} chips that spread over the available bandwidth for the communication. A large value of channel bandwidth helps to achieve a high data rate but at the expense of more noise. For the up-link communication, the LoRa packet's preamble is modulated by up-chirps while down-chirps modulate the payload, and vice versa for the down-link communication.

I.3.4.4 Transmission Power (TP)

A regulating parameter for LoRaWAN communication is Transmission Power (TP). Transmission power for LoRa networks ranges from 2 dBm to 20 dBm [22] (+14 dBm in Europe). The communication range is extended by using larger TP values, but the power consumption goes up. An end device's utilization of transmission power can interfere with other end devices' transmissions.

I.3.4.5 Coding Rate (CR)

The degree of redundancy employed by the Forward Error Correction (FEC) to detect errors in each data transfer is known as the Coding Rate (CR). In order to do this, 4-bit redundancy data is encoded into (4+n) -bits, where $n \in \{1, 2, 3, 4\}$. It indicates that 5, 6, 7 or 8 transmission bits are used to encode every four usable bits. The higher the data transmission duration on air, the lower the coding rate. Generally, the code rate for LoRaWAN protocol is defined as $\left(\frac{4}{4+n}\right)$ [23].

The parameters SF, CR, and BW are use to calculate the bit rate (R_b) [23] for LoRa modulation

$$R_b = SF^* \frac{BW}{2^{SF}} * CR \qquad [bit/s] \tag{I.1}$$

Using this equation, the data rates can be calculated for all spreading factors.
where:

- SF = Spreading Factor (7, 8, 9, 10, 11, 12).
- CR = Code Rate (1, 2, 3, 4).
- BW = BandWidh in Khz (125, 250, 300).
- R_b = Data Rate or Bite Rate in bps.

I.3.4.6 Symbol transmission time

The Spreading Factor determines the T_{symbol} transmission time for each symbol in LoRa. As the SF increases, so does the transmission time. For the same bandwidth, the transmission time of a symbol in $SF8$ is twice that of a symbol in $SF7$. This is illustrated in Fig.I.5 and is true for up to $SF12$.

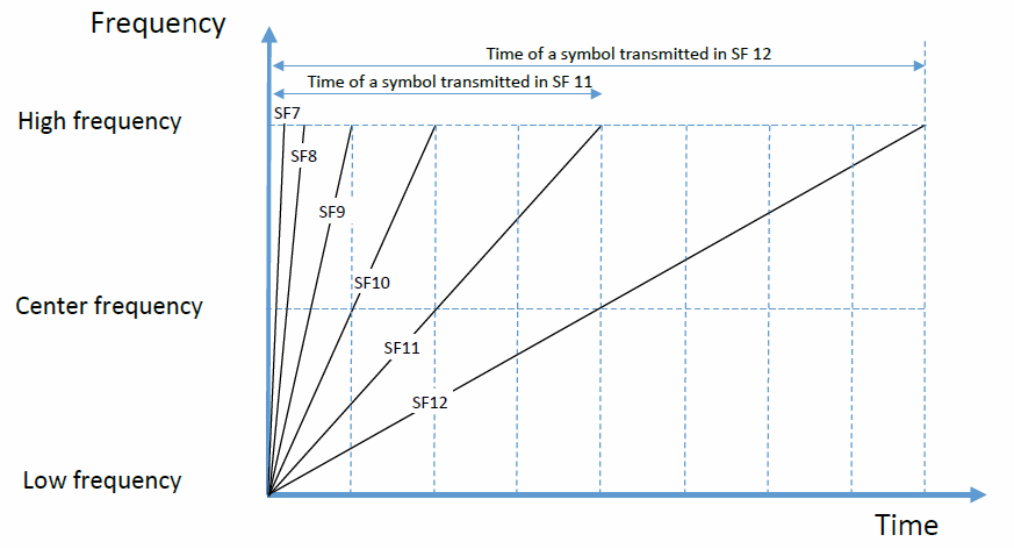


Fig. I.5 – *Symbol transmission time [16].*

The bandwidth utilized also affects each symbol's transmission time T_{symbol}. The bandwidth is inversely proportional to T_{symbol}. The following expression [24] is obtained considering the SF and the bandwidth:

$$T_{symbol}(SF) = \frac{2^{SF}}{BW} \tag{I.2}$$

As an example, Table I.4 shows the transmission time depending on SF, for a bandwidth of 125 KHz.

Table I.4 – *Symbol transmission time for BW125 [18].*

Spreading Factor (SF)	Symbol transmission time (ms)
SF7	1.024 ms
SF8	2.048 ms
SF9	4.096 ms
SF10	8.192 ms
SF11	16.384 ms
SF12	32.768 ms

Obviously, the higher the bandwidth, the higher the symbol rate.

I.3.4.7 Time On Air (ToA)

The Time on Air is the overall duration of a LoRa frame. The calculation of ToA needs the combination of spreading factor, bandwidth, coding rate and the packet structure.

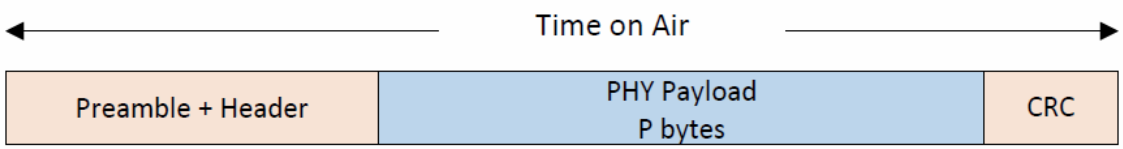


Fig. I.6 – *Time on Air of a LoRa frame [16].*

The bit rate has a direct impact on the ToA of transmissions, which is the time T_{symbol} needed to transmit a packet on the wireless channel, so the time taken to send 2^{SF} chips at the

chip rate and it is computed with the equations I.2.

The preamble duration depends on the number of preamble symbols $n_{preambule}$ and T_{symbol} as follows:

$$T_{preamble}(SF) = \left(n_{preamble} + 4.25\right) * T_{symbol}(SF) \tag{I.3}$$

where $n_{preambule}$ is the number of programmed preamble symbols , 8 symbols according to EU868 [3] , and T_{symbol} being a symbol's duration.

The payload duration depends on the header mode and the low data rate optimization. The number of symbols $n_{payload}$ that comprise the header and the packet payload is calculated as following Equation [8]:

$$n_{payload} = 8 + \max\left(\text{ ceil } \left[\frac{8PL - 4SF + 28 + 16 - 20IH}{4(SF - 2DE)}\right] * (CR + 4), 0\right) \tag{I.4}$$

where:

- max = the maximum operator, selecting the highest value between the two inputs.
- PL represents the payload size in Bytes.
- IH is the indicator of whether header is enabled $(IH = 0$, otherwise $IH = 1)$.
- DE is the indicator of whether the Low Data Rate Optimize is enabled $(DE = 1$, otherwise $DE = 0)$.
- CR is the coding rate from 1 to 4

Then, we can deduce the payload duration as follows:

$$T_{payload}(SF) = n_{payload} * T_{symbol}(SF) \tag{I.5}$$

The time needed to send a LoRa frame from an end device to the gateway, given BW, SF, and CR, is equal to the sum of the preamble's transmission time ($T_{preambule}$) and the payload's duration ($T_{payload}$), which is:

$$ToA(SF) = T_{preambule}(SF) + T_{payload}(SF) \tag{I.6}$$

where, $T_{preambule}$ is the preamble duration and $T_{payload}$ is the payload duration.

I.4 LoRa robustness

LoRaWAN uses LoRa Chirp Spread Spectrum (CSS) modulation, which is very resistant to interference and noise, and it is less affected by fading effects (signal weakening) than other technologies; which allows for better data availability in difficult signal propagation conditions. This is a technology that modulates signals in the Industrial, Scientific and Medical (ISM) band, it uses a frequency band of 868 MHz in Europe and 915 MHz in the United States. LoRa communication can operate on a bandwidth $\in \{125, 250, 500\}$ kHz depending on the regional settings.

I.4.1 Lora robustness to noise and interference

Based on CSS modulation which uses the chirp technique whose signals have a constant amplitude with a variable frequency. The evolution of the frequency of such signal is always strictly increasing UpChirp or decreasing DownChirp and linear over time [25]. As there is a spread spectrum, this makes the transmission very resistant to interference and multipath effects, as illustrated in Fig.I.7.

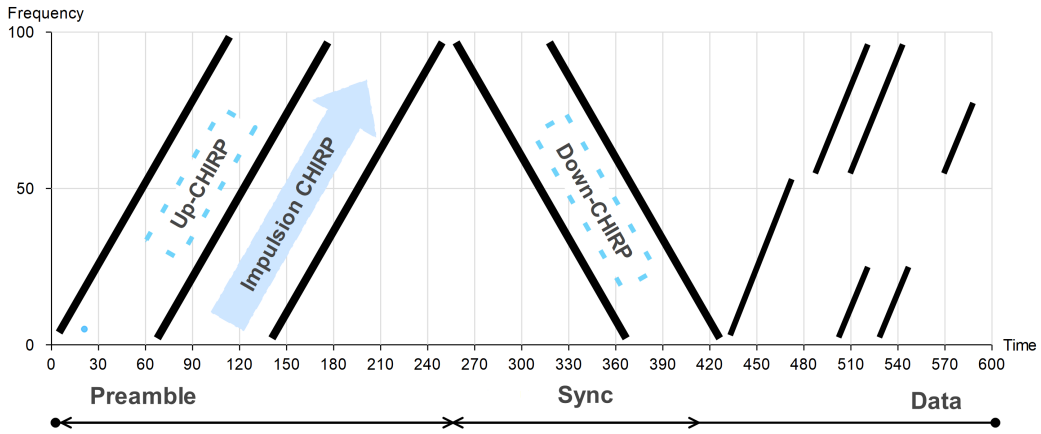


Fig. I.7 – *Lora Chirp Spread spectrum modulation*

An adjustable number of coded bits, known as the Spreading Factor (SF) is used to represent a symbol with a chirp's initial frequency; thus, a Chirp employs an SF that represents 2^{SF} bits per symbol. CSS modulation allows the signal energy to be spread over a wider band based on SF. On a more technical level and in accordance with LoRaWAN specifications [26], here are some useful quantitative parameters to characterize a transmission:

- Being a frequency bandwidth (BW) of the signal BW, the SF allows the symbol duration T_{symbol} to be determined as described in equation I.2.

- A longer transmission time (symbol time) makes the message more resilient to noise or interference. However, the fact that there are more symbols for higher SFs, which increase the frequency of symbol errors, can somewhat offset this benefit. The receiver sensitivity is defined by :

$$S = -174 + 10log_{10}(BW) + NF + SNR \quad [dBm] \tag{I.7}$$

where (-174) represents the thermal noise of the receiver, NF is the noise figure of the receiver (which is fixed for a hardware configuration data) and SNR is the signal-to-noise ratio required for modulation.

Since the several etalement variables are theoretically orthogonal to one another, one of the primary advantages of lora modulation is that the receiver's demodulator can differentiate between concurrent diffusions entering the same frequency channel. This pseudo-orthogonality between different packets allows a lora network to exploit different SFs to achieve higher throughput compared to conventional modulation systems.

I.4.2 Lora robustness to attack

The LoRaWAN node and the network establish mutual authentication through a network connection mechanism. By guaranteeing that only legitimate and approved devices are able to connect to the network, this helps to prevent spoofing attempts and offers further security. LoRaWAN has significantly evolved to address security threats, introducing robust encryption mechanisms, improved key management, and protections against common attacks over several releases. The first version of LoRaWAN (v1.0) introduced the foundations of AES-128 bits

encryption security combined with different operating modes, such as: the messages are encrypted with AES-128 bits in Counter Mode (CTR) to ensure confidentiality. This ensures the confidentiality of the data and cannot be accessed by unauthorized third parties. Using AES-Cipher-based Message Authentication Code (AES-CMAC) to verify the integrity of messages, it is a signature that prevents any unauthorized alteration or modification of the data. This ensures the robustness and the non-repudiation property as will be detailed in the last chapter of this thesis.

I.5 LoRaWAN

A network protocol called LoRaWAN makes use of the LoRa modulation scheme. Thus, considering their entwined relationship, they are two distinct entities. The LoRa Alliance is a non-profit, open organization that was established in 2015 by prominent technology industrialists. Its objective is to standardize low-power wide-area networks in order to deploy industrial application technologies, machine-to-machine (M2M), and the Internet of Things. Through gateways, it permits communication between end devices and a network server. The LoRaWAN protocol stack consists of three levels. : the physical layer, the data link layer, and the application layer, as shown in Fig.I.3. The data that is transferred across the network is handled by the application layer. The LoRaWAN protocol, which is outlined by the LoRa Alliance, is implemented at the data link layer. The physical layer implements the LoRa radio frequency technology from Semtech [3]. Data rates for Europe correlate to a metric known as DR and range from 0.3 kbps to 50 kbps. The channel's bandwidth, which is equivalent to 125 kHz for DR0 to DR5, is SF12. For a given transmission, the bandwidth and spreading factor utilized by the underlying LoRa modulation determine the data rate of LoRaWAN. The attainable bit rate is always increased with a decrease in SF or an increase in BW. Table I.5 [19] illustrates this.

Table I.5 – *Relationship between SF, DR and ToA widh a 125 kHz a 10 Byte Packet.*

Data Rate (DR)	Spreading Factor (SF)	SNR Limit	Time-on-Air (ToA)	Bit Rate (bit/s)
5	7	-7.5	56 ms	5470
4	8	-10.0	103 ms	3125
3	9	-12.5	205 ms	1760
2	10	-15.0	371 ms	980
1	11	-17.5	741 ms	440
0	12	-20.0	1.483 ms	250

I.5.1 LoraWan Version and History

In light of the latest development in resource-constrained IoT devices, To enhance LoRaWAN's performance in terms of security, scalability, and real-time long-range communication, several versions have been released. The Grenobloise firm Cycléo created LoRa technology, which was later patented and sold by Semtech, a company that makes LoRa chips.

In January 2015, the LoRaWAN v.1.0 protocol, also referred to as the draft version, saw the release of its first stable version. The LoRa alliance issued a few minor revisions in the next years. Released in February 2016, version 1.0.1. The clarification of the *RX* window start time definition, the requirement to use a coding rate of 4/5 to provide a maximum time-on-air < 400 ms, and the adjustment of the *JoinAccept MIC* (The message integrity code) calculation were the primary modifications included in this version. Moreover, two unique keys *NwkSKey* and *AppSKey* that are both descended from a single root key were introduced with the release of LoRaWAN v1.0.2 in July 2016. This specification resulted in the improvement and separation of the PHY layer portion that described the regional parameters into a new document ("LoRaWAN Regional Parameters" [27]). In order to prevent replay assaults, LoRaWAN v1.0.2

suggested solutions that involved encrypting the uplink frame counter (*FCntUp*) and utilizing it in the downlink confirmation messages (*ACKdownlinks*). LoRaWAN v1.0.3, which added full support for unicast and multicast Class B devices and made several drastic changes to address vulnerabilities in earlier versions, and LoRaWAN v1.1, which allows handover roaming and improves security, are the two most significant improvements.

To prevent replay attacks, *JoinAccept MIC* handles security enhancements in the Over-the-Air Activation (*OTAA*) method. Additionally, the Join server (JS) logical entity is introduced, which is in charge of managing the ED authentication process. Moreover, *ReJoin-Request* messages and new session keys are introduced. LoRa nodes can now take advantage of numerous Network Servers thanks to a major feature added in version 1.1: roaming. The LoRa Alliance has released the LoRa Backend Interfaces definition paper in addition to the protocol definition. The document covers Over The Air Activation and Roaming procedures of end-nodes because the roaming process involves numerous network servers. Passive and Handover Roaming for ongoing sessions are two examples of this. In October 2020, LoRaWAN v1.0.4, the most recent protocol specs for new LoRaWAN improvements, was released [28].

A summary of the LoRaWAN specifications' development, including the year of release and significant modifications, is shown below.

1. LoRaWAN version 1.0 (January 2015) [29]: The first approved version of LoRaWAN 1.0.
2. LoRaWAN version 1.0.1 (February 2016)[30] : This small update added additional frequency plans, changed certain MAC instructions, and clarified a few issues that were unclear in the prior version.
3. LoRaWAN version 1.0.2 (July 2016) [27]: The "Physical Layer" Section was transferred to a new document, several errors were rectified, and more MAC commands were added in this minor edition.
4. LoRaWAN version 1.1 (October 2017) [3]: With the addition of a new server named Join Server (JS), this significant revision introduced a new architecture. It also brought many improvements to the security mechanism, including support for roaming handover and the use of two root keys rather than one to derive the session security keys. Finally, it added numerous countermeasures to mitigate some of the vulnerabilities that had been reported in earlier versions. This thesis includes the following specifications:
    - **Several FCntDownCounters:** are supported in the LoRaWAN v1.1, as opposed to just one *FCntDown* counter that is shared across all ports in LoRaWAN v1.0.x (i.e. 1.0.1 and 1.0.2).
    - **Reset indication commands:** (i.e *ResetInd*, *ResetConf*) LoRaWAN v1.1 ABP devices are the only ones with Reset indication commands available. This MAC command is not supported by LoRaWAN 1.0.x servers.
    - **Rekey indication commands:** Rekey indication commands: (i.e *RekeyInd*, *RekeyConf*) are only available to LoRaWAN v1.1 OTA devices. LoRaWAN 1.0.x servers do not support this MAC command.
    - **Device time commands:** (i.e *DeviceTimeReq*, *DeviceTimeAns*) are only available to LoRaWAN v1.1 activated devices. LoRaWAN 1.0.x servers do not support this MAC command.
    - **Application session key:** Only the *AppKey* in LoRaWAN v1.1 network servers is used, not the device's *AppKey*.
    - **Device time request/answer command:** The conventional *BeaconTimingReq* & *BeaconTimingAns MAC* instructions from earlier LoRaWAN versions are replaced by this in LoRaWAN v1.1.

5. LoRaWAN version 1.0.3 (July 2018) [26]: as the most recent version, which fixed some of the performance problems with LoRaWAN v1.1 for class B devices and added a few *MAC* instructions for class A devices, surpassing and improving upon earlier versions (LoRaWAN v1.0.1 and v1.0.2). As a result, its numerous requirements are presented as follows:

    - **Device time commands:** DeviceTimeReq and DeviceTimeAns are *MAC* commands that are only accessible and enabled on LoRaWAN v1.0.3; they are not available on LoRaWAN v1.0.1 or v1.0.2.
    - **Beacon timing request/answer commands:** are also *MAC* commands that are only usable with LoRaWAN v1.0.3. The device can also use another command (*DeviceTimeReq* & *DeviceTimeAns*) as a stand-in.
    - **Unicast/Multicast Support:** is added in LoRaWAN v1.0.3 to support class B devices.
    - **Device time request command:** Additionally, this *MAC* command was included as a new time synchronization functionality for class A and C devices in LoRaWAN v1.0.3.
    - **Compatibility:** LoRaWAN v1.1 class B devices are compatible with LoRaWAN v1.0.3 class B devices.

6. LoRaWAN version 1.0.4 (October 2020) [31] : The joining channel selection procedure, retransmission backoff, *ADR* behavior, and *FCnt* usage and behaviors were all clarified by this minor update for v1.0.3. The two major security changes we noticed in this release are that *AppNonce* and *JoinEUI* have been replaced with *AppEUI* and *AppNonce*, and that *DevNonce* generation is now incremental instead of random.

I.5.2 LoRaWAN Architecture

In a LoRaWAN architecture, messages are relayed between end devices and the network server via one or more LoRa gateways, creating a network topology star. Gateways serve as a bridge between RF transmissions from LoRa and IP packets, connecting them to the Network Server via IP protocol. A simple LoRaWAN architecture is shown in Fig.I.8, where end-devices broadcasts its data to each gateway located in its vicinity , the gateways transfer this packet to the network server. The network server collects messages from all gateways and filters duplicate data and determines the gateway which has the best reception; the gateways only serve as relays to reach the server managing the network, itself connected to one or more application servers. The packets sent by the end-devices are retransmitted by the gateways after only adding information concerning the quality of the received signal; the network server forwards the packet to the appropriate application server where the end user can process the data from the sensor. The application server can send a response back to the end node; when a response is sent, the network server receives the response and determines which gateway to use to broadcast the response to the end node [32].

The functionality of each entity is as follows:

- **End Device (End Node)**

LoRaWAN devices are embedded electronic systems belonging to the IoT world: low energy consumption, small size, low power and low cost; designed to collect data from physical environment and transmit it efficiently to gateways using single-hop LoRa communication [33]. There is a distinction between uplink and downlink messages in LoRaWAN nomenclature. End devices send uplink messages to the Network Server through one or more gateways, while the Network Server sends downlink messages to a single end device via a single gateway.

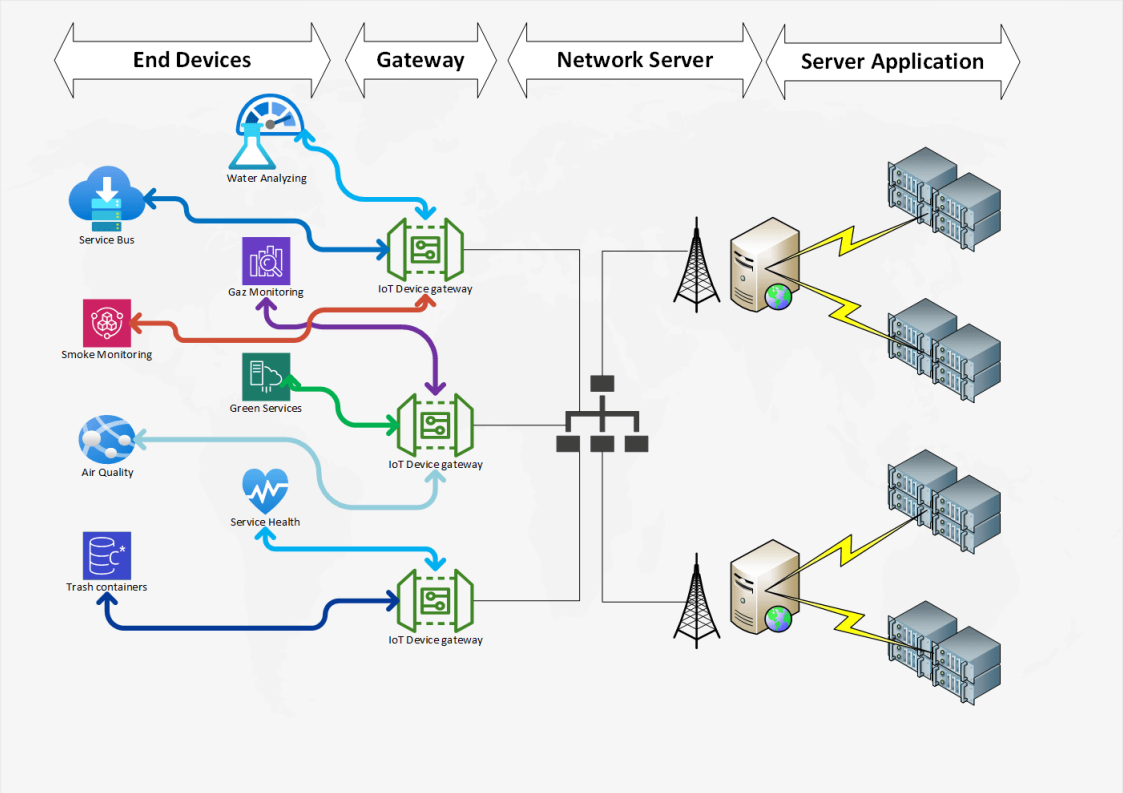


Fig. I.8 – *LoRaWAN Architecture.*

- **GetWay (GW)**

A baseband processor and a LoRa transceiver chip make up their construction. They accommodate eight to sixty-four channels and are capable of decoding several channels at once. The LoRaWAN Network Server and the sensors are connected by LoRaWAN® gateways. They receive and send the data sent by the sensors to the LoRaWAN Network Server using the low-power, long-range LoRaWAN® protocol.They are essential for expanding the network's coverage since they are able to receive signals from sensors located throughout a large area. Every LoRaWAN gateway is uniquely identified by its 64-bit EUI. On a network server, this identity is helpful for registering and activating a gateway.

➤ **In Uplink**

- ✓ They listen on every possible channel. That is to say on all frequencies (frequency carrier), bandwidths (BW) and Spreading Factor (SF) authorized or possible.
- ✓ An end-device broadcasts its data to every gateway in its neighborhood. The packets sent by the end-devices are retransmitted by the gateways, after only adding information about the quality of the signal received.
- ✓ The gateways forward this packet to the network server configured in the gateway beforehand.

➤ **In Downlink**

- ✓ The application server can send the end-device a downlink (either an acknowledgement or a command action).
- ✓ The role of the network server is to determine which gateway to use to broadcast the response to the end device.
- ✓ A LoRa-modulated radio communication is received by the Gateway, which then sends an IP frame to the Network Server.

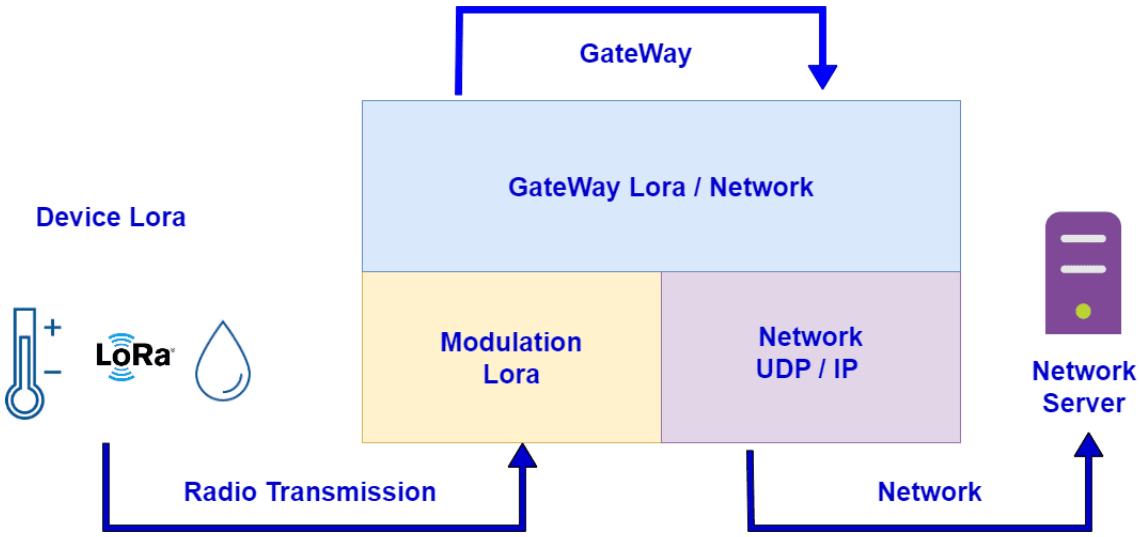


Fig. I.9 – *Role of Lora Gateway* .

Radio interface side : The Gateway receives a LoRaWAN frame and extracts the PHY Payload. It will encode this PHY Payload in ASCII format in base 64. The Gateway also extracts all the useful information on the characteristics of the reception that took place: SF, Bandwidth, RSSI, Time On Air. . . . . .

IP network interface side: The Gateway transmits all of this information in an IP packet (UDP) to the Network Server. The data transmitted is text in JSON format. The Gateway therefore has a role as a gateway between the LoRa protocol on one side and an IP network on the other.

- **Network Server (NS)**

Is the LoRaWAN network's most intelligent component [34]. The entire LoRaWAN® infrastructure must be managed by the LoRaWAN® Network Server. After receiving the data packets that the gateways have passed, it carries out operations like routing, authentication, and decryption. By locating the relevant application server, the network server makes sure the data gets to its destination. Additionally, it oversees the network's security features, such as managing encryption keys and device registration and activation.

Data routing, security, and authentication are the principal responsibilities. Only authorized devices can join to the network through the Network Server's authentication and authorization processes. It also manages data encryption and decryption, protecting private data while it's being transmitted. The Network Server also reduces latency and maximizes network resources. The Network Server makes it possible for dependable and secure IoT communication at scale by efficiently handling these responsibilities.

- **Application Server (AS)**

A crucial element that communicates with the LoRAWAN Network Server and allows for the processing, administration, and safe interpretation of the data gathered is the Application Server. It handles data in accordance with particular application needs after receiving it from the LoRaWAN network server. In addition to creating all downlink payloads from the application layer to linked end devices, this server is in charge of storing and evaluating the sensor data that is received, causing pertinent actions or notifications. A network server sends encrypted messages to the application server. AppSKey (Application Session Key), a 128-bit AES key, is used for encryption and decryption. This procedure will be covered in full in Chapter IV.

- **Join Server (JS)**

The Join Server helps with session key generation, root key storage, and secure device activation. By transmitting the Join-request message to the Join Server via the Network Server, the end device starts the join process. The Join-server creates session keys, analyzes the Join-request message, and sends *NwkSKey* and *AppSKey* to the Network server and the Application server, respectively. With LoRaWAN v1.1, the Join Server was first made available. Additionally, LoRaWAN v1.0.4 has it. The Fig.I.8 does not display the Join Server.

I.5.3 LoRa node classes

Depending on when they plan the reception of downlink communication, end-devices in LoRaWAN can be set to function in accordance with one of three classes. For end-devices, LoRaWAN has three classes of operation: class A (for All), class B (for Beacon), and class C (for Continuously listening) [35], as shown in Fig. I.10.

All devices classes support bidirectional communication (uplink and downlink).

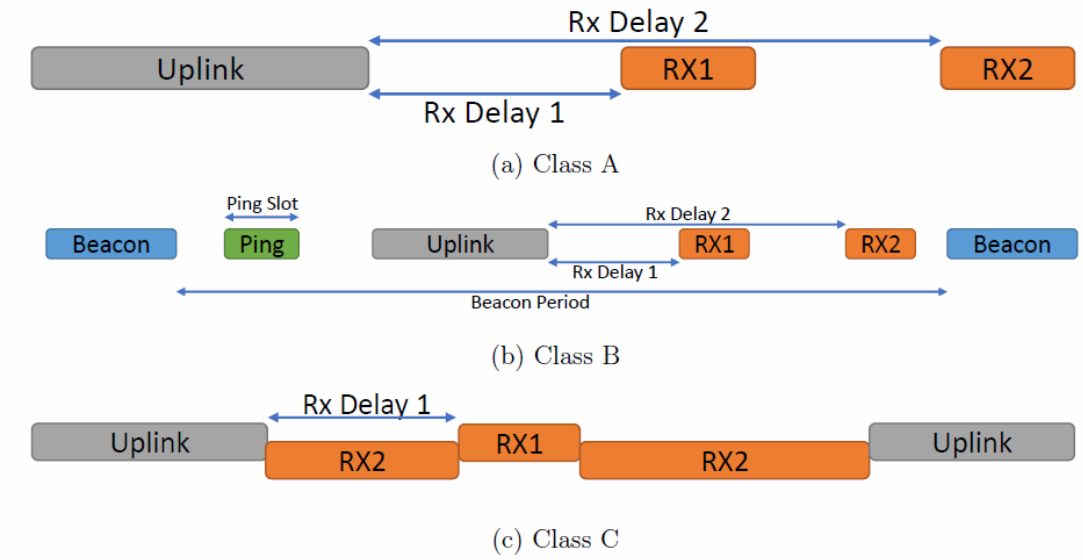


Fig. I.10 – *LoRaWAN devices classes.*

I.5.3.1 Classe A (All)

Each LoRaWAN end device must support Class A implementation. The end devices allow bi-directional communications, with the acknowledgments (*ACK*) from each end device being received during two short down-link receive windows named *RX1* and *RX2*. Each end device's up-link transmission is followed by these two windows. Only in the event that *RX1* is unsuccessful in receiving an *ACK* is *RX2* initiated.

RX1 opens after a delay called *RECEIVE_DELAY1* (in seconds) and *RX2* opens exactly 1 second after the first one opens. Put otherwise, the end-device opens *RX2* after waiting a second after *RX1* closes. That implies:

$$RECEIVE\_DELAY2 = RECEIVE\_DELAY1 + 1sec$$

as shown in Fig.I.10. As stated previously, if no *ACK* is received during *RX1*, the end-device listens for possible *ACKs* during *RX2*. If no *ACK* is received during *RX2*, then the up-link packet is considered lost and the end-device transmits it again. The maximum number of re-transmissions in LoRaWAN is set to 8 attempts by default [19].

The delay between two transmissions has to be larger than or equal to 99 times the duration of the frame transmission in order to respect the duty cycle of 1%.

Class A end devices use incredibly low amounts of power. They can therefore run on battery power. They typically have significant periods between uplinks and spend the majority of their time in sleep mode. Class A devices also have a high downlink latency since they have to send an uplink in order to receive a downlink.

Receive window parameters: The DR to be used in $Rx1$ is set as the uplink DR minus an offset called $RX1DROffset$. $RX1DROffset$ can take values in the range of 0 to 5. The DR for the first receive window is by default the same as the one used in the previous uplink transmission because $RX1DROffset$ has a default value of zero. The initial receive window utilizes the same frequency channel as the previous uplink transmission. By default, DR0 and 869.525 MHz are the fixed data rate and frequency configurations used in the second receive window.

I.5.3.2 Classe B (Beacon)

When ping slots, or receive windows, are opened on a regular basis to receive downlink communications, Class B devices expand the capabilities of Class A devices. Periodically, the end devices receive a time-synchronized beacon (unicast or multicast) that is broadcast by the network through the gateways. For the purpose of synchronizing the end devices' internal clocks with the network, these beacons supply a timing reference. The network server can then determine when to deliver a downlink to a particular device or collection of devices. Beacon period: The interval of time between two beacons.

Similar to Class A devices, the two short receive windows, *RX1* and *RX2*, will open following an uplink. Compared to Class A end devices, Class B end devices have lower downlink latency since they ping slots on a regular basis. However, compared to Class C end devices, they have a far higher latency. Batteries are usually used to power Class B devices. The battery life of Class B devices is lower than that of Class A devices because they respond to beacons and have free ping slots for longer periods of time.

I.5.3.3 Classe C (Continious)

As seen in the Fig.I.10, Class C devices increase the capabilities of Class A devices by maintaining open receive windows unless they are delivering an uplink. As a result, Class C devices have extremely low downlink latency since they can receive downlink messages practically always. Class C devices offer the lowest latency when compared to Class A and Class B devices [36]. However, because they must open continuous receive slots, they use more power. These devices are consequently frequently mains powered because they cannot be run on batteries for extended periods of time.

The properties of the LoRaWAN class are summarized in Table I.6. As previously stated, in the case of Class A, the sensor initiates all transactions, whereas the network server is limited to transmitting two downlink slots. After that, until the downlink frame is demodulated, the radio receiver remains operational. If a frame was detected and demodulated within the first receive window and was intended for this end-device after address and MIC (message integrity code) tests, the end-device does not open the second receive window. This class has the least effect on the sensor's battery life. For all these reasons, the remainder of this thesis will examine LoRaWAN Class A.

Table I.6 – *LoRaWAN class comparison.*

Lora Class A	**Lora Class B**	**Lora Class C**
Battery Powered	Low Latency	No Latency
Bidirectional Communications	Bidirectional with scheduled receive slots	Bidirectional communications
Unicast Messages	Unicast and Multicast Messages	Unicast and Multicast Messages
Small Payloads, long intervals	Small Payloads, long intervals, Periodic beacon from Gateway	Small Payloads
End Device initiates communication (Uplink)	Extra receive window (Ping Slot)	Server can initiate transmission at any time
Server communicates with End-Device (downlink) during predetermined response windows	Server can initiate transmission at fixed intervals	End-Device is constantly receiving

I.5.4 Duty Cycle limitations

Certain limitations are placed on accessing the physical medium by the European Telecommunications Standards Institute ($ETSI$) standards. These limitations vary by nation/region and include limitations on the maximum transmission power, duty cycle, and maximum time a transmitter can transmit in an hour. One may choose to use a Listen Before Talk (LBT) transmissions management system or a duty cycle limiter under the $ETSI$ requirements.

According to [3] specifications, the imposed duty cycle in the EU for the regularly utilized frequency sub-bands of 863.00 – 868.00 MHz and 868.00 – 868.60 MHz is 1%. The end device's maximum message transmission rate is restricted by LoRaWAN since it only employs duty-cycled transmissions. Let d denote the duty cycle in a certain sub-band in this manner. Then, there must be a minimum off-period equal to the duration of which the channel is unavailable for transmission by this node after the time on air T_a required to transmitted a packet in this sub-band.

$$T_{off} = T_a(\frac{1}{d} - 1)seconds \tag{I.8}$$

where:
T_{off} = time of silence required after transmission.
T_a = Time on Air.
d = duty cycle.

Thus the maximum duty cycle is the maximum percentage of time during which an end-device can occupy a channel.

Considering a device transmitting on a channel with a 1% duty cycle, we can show how a duty cycle constraint corresponds to a specific maximum duration on air and minimum waiting time between consecutive packet transmissions. 10 transmissions lasting 3.6 seconds can be completed by this device in an hour. It's possible for the device to continue transmitting on a different sub-band during the period when that sub-band is unavailable. The device must wait to transmit again if any sub-bands are unavailable. The default channels and duty cycle limitations for LoRaWAN in Europe are displayed in Table I.7.

Since the designer determines the transmission interval between packets $(Tx_{interval})$, potential transmissions that exceed duty-cycle restrictions are typically not regarded as outage conditions. However, if the transmission parameters change for any reason, like when an ADR algorithm is activated, a device may attempt to send more than its duty cycle limit. Using DR5 (SF7 and BW125), for instance, a device configured to send a packet of 10 bytes every 90

Table I.7 – *LoRaWAN default channels and duty cycle limitations in Europe.*

Frequency (MHz)	Direction	Duty Cycle (%)	Max transmission power (dBm)
868.1	DL, UL	1 %	14
868.3	DL, UL	1 %	14
868.5	DL, UL	1 %	14
869.525	DL	10 %	27

seconds on a single sub-band functions as planned with an air-time of 61 ms and T_{off} of 6.1 seconds (from Equations I.8). The identical device would have a ToA of 1482.8 ms and a T_{off} of 146.8 seconds, exceeding 1% duty cycle, if it were to convert to DR0 (SF12 and BW125) without changing its $(Tx_{interval})$. Since LoRa transceivers are typically configured to adhere to this limit, the chip itself would halt the transmission [37]. This leads to "invisible" packet loss because the node's frame-counter alone would not reveal this loss. Unless the device's behavior has been specifically engineered to do so, the data will not be queued up for retransmission, and the packet's contents will be lost. Although it is possible to manually override this behavior, doing so might be against the spectrum access laws.

I.5.4.1 LoRa Network Collision Effect

The basis of LoRaWAN is a pure ALOHA channel access technique. In its most basic form, this enables data-sending devices to send data whenever they want, without having to worry about schedule conflicts, availability, or even a successful connection. Consequently, collision frames arising from concurrent transmissions may likely be lost or damaged.

On the other hand, private LoRa networks can be established by utilizing unlicensed ISM frequencies. However, when the number of LoRa devices increases greatly, the network's capacity will saturate and performance will suffer. The fact that every nearby gateway will receive packets from any LoRa device, irrespective of their network provider, exacerbates this issue by creating inter-network interference. Collisions and packet loss are caused by this interference. When packets use the same BW, SF, and carrier frequency and overlap in time, a collision occurs in LoRa.

The probability of collisions depends on the selection of LoRa parameters used to transmit a packet. When several devices use the same configuration, the probability of occurring collisions is high. Furthermore, the selection of Tx and SF influences the coverage, some packets do not collide due to the signal attenuation over distance. In addition, traffic volume, transmission frequency, and packet size all influence the probability of a collision. Higher time on air and channel occupancy are caused by more frequent transmissions and larger packet sizes.

LoRa signals and non-LoRa signals are the two sources of interference. For SF = 7 with an error correcting scheme of 4/6, it was estimated [38] that a single tone pulse is not an issue if it is less than 6 dB above the target signal. For the second category, the LoRaWAN physical layer supports adjustable SFs, $SFs \in [7-12]$ and spread spectrum signals with different SFs have good orthogonality and can transmit in the same channel simultaneous without interference [39].

Table I.8 presents co-channel rejection for all combinations of the desired signal SF_d and the interferer signal SF_i [40]. These values define the interference model. In the Table I.8, each element represents the minimum signal power margin threshold $V_{i,j}$, with $i, j \in [7, ..., 12]$, that a packet transmitted with $SF = i$ must have in order to successfully decode over any interfering packet with $SF = j$. If, after taking into account all possible combinations of SF, a higher power margin value (dB) than the associated co-channel rejection value is fulfilled, the packet survives interference with all interfering packets. One thing to observe is that the matrix's values are not symmetric as intercepting another packet with a lower SF requires more power to decode. This is due to the fact that the signal power increases with decreasing SF configuration. If a packet equipped with $SF8$ intercepts another prepared with $SF10$ (30

dB), for instance, a higher power is required to decode the packets. If the converse occurs, however, a lower power margin value (22 dB) will be required in order to decode the $SF8$ packet that the $SF10$ packet intercepted.

Table I.8 – *Co-channel rejection [dB] for all combinations of SFs for the desired and interferer packets.*

Desired Packet \ Interferer Packet	$SF7$	$SF8$	$SF9$	$SF10$	$SF11$	$SF12$
$SF7$	-6	16	18	19	19	20
$SF8$	24	-6	20	22	22	22
$SF9$	27	27	-6	23	25	25
$SF10$	30	30	30	-6	26	28
$SF11$	33	33	33	33	-6	29
$SF12$	36	36	36	36	36	-6

I.5.4.2 Intra-SF and inter-SF Interference:

When two packets are received on the same channel with the identical SF configuration, LoRa GWs cannot decode them. Because of co-SF interference, this method causes both packets to be lost. However, inter-SF collisions, which are known to result in packet loss, occur when two packets with different SFs are received simultaneously on the same channel.

- **Intra-SF collisions:** occurs when the colliding frames are of the same SF. The frame with the highest power will be decoded if it is at least 6 dB higher than the other LoRa frame: $RSSI_d$ - $RSSI_i$ $\geq$6dB.
- **Inter-SF collisions:** occurs when the colliding frames are of different SF_s ($SF_d \neq$SF$_i$). The frame is demodulated if the power difference is strictly greater than the inter-SF collision threshold which depends on the SF of the corresponding frame (Table I.8): frame "d" is demodulated if: $RSSI_d$ - $RSSI_i$ > Thr ($RSSI_d$).

As shown in Fig.I.11, in LoRaWAN , it is assumed that the available SFs are SF7 and SF12, and the following three cases are transmitted in the channel. In case 1, on account of two SFs12 arrive at different channels simultaneously, they can be successfully received and decoded. In the case 2, SF7 and SF12 are in the same channel, and due to their orthogonality, collisions can be avoided. In the case 3, the same SF7 arrives at the same channel simultaneously, causing a collision to occur. When there are multiple SFs transmitting on the channel, two collide occur: Two packets with same SF arriving at the same channel simultaneous cause a collision, thus causing data packet loss.

However, it has been proven that the quasi-orthogonality of the different SFs can have a big impact on the overall network quality and scalability, with each packet also experiencing interference from signals sent at the same time, on the same frequency, but on a different SF [41]. Because of the capture effect, a packet can still be received over this interference if it achieves a minimum power difference from the strongest interferer. These thresholds have been modelled according to the work of Reference [41]. They show how a packet will be received over another on the same SF if the first is at least 6 dB stronger than the second.

More recent research conducted by Reference [42] shows that different thresholds are obtained with an experimental setup. Here, only 1 dB separation is needed for the strongest packet to be correctly received over the total interference on the same SF. The overall chance of packet collision is affected by the nodes' location and density, as well as their SF, packet transmission interval, and packet air time.

In conclusion, it is possible to assume that a collision will happen when two or more data packets share the same carrier frequency, use the same SF, and have overlapping arrival timings.

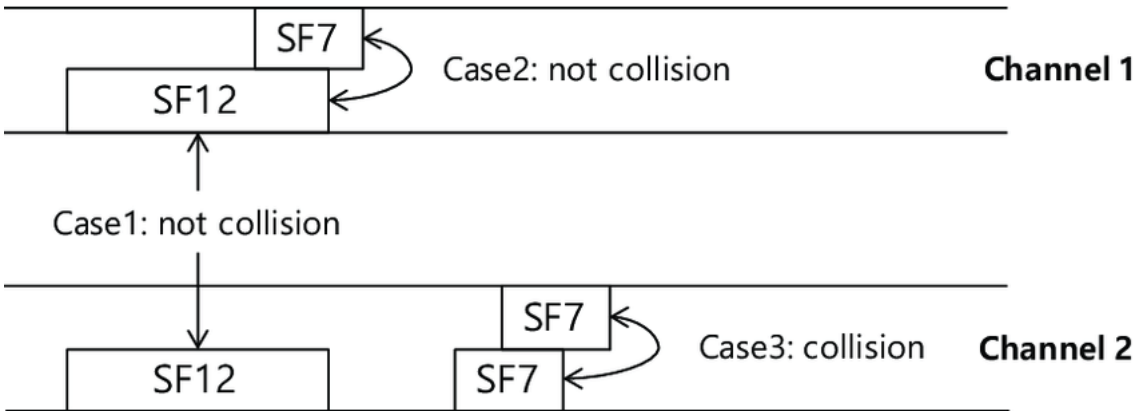


Fig. I.11 – *Multiple access under the orthogonal SF.*

Additionally, in the event of a collision, a capture effect will take place, whereby the data packet with the greater power is received and the other is lost, provided that the difference between the received power of the two packets is at least 6dB. When the difference is less than 6 dB, the receiver can't effectively decode either packet and will continuously flip between them.

I.5.5 Acknowledgement and Retransmission Procedures

LoRaWAN makes a distinction between two kinds of frames: confirmed frames, which the recipient must acknowledge, and unconfirmed frames, which the recipient must not. Transmissions in the uplink direction can be verified or unverified. Unconfirmed frames are only sent once, and the network server is not supposed to acknowledge them. This implies that the end device can send the subsequent frame independently upon receiving the preceding frame in the event that an unconfirmed frame is lost.

The network server is supposed to recognize confirmed frames. One of the two receive windows ($RX1$ and $RX2$) that immediately follow the broadcast is when the end device anticipates receiving a downlink acknowledgment (ACK). The end device repeats the message until either a ACK is received or the message's maximum number of transmission attempts is achieved, which is eight by default, if the ACK is not received.

If the sender is an end-device, the network uses one of the receive windows that the end-device opens following the send operation to send the ACK through one of the gateways within range. The end-device sends a ACK at its own choice if the sender is a gateway. Keep in mind that $ACKs$ are never retransmitted; they are only sent in response to the most recent message received. Additionally, according to the LoRaWAN standard [3], end devices must wait $ACK\_TIMEOUT$ seconds before retransmitting when confirmed traffic is used. $ACK\_TIMEOUT$ is a random delay value that ranges from 1 to 3 seconds. Additionally, just like any other regular transmission, the retransmission must adhere to the duty cycle limitation.

While $ACKs$ communicated on $Rx1$ should use the same SF as the UL transmission, $ACKs$ transmitted on $Rx2$ should utilize the lowest available data rate (i.e. [SF = 12, DRo]), according to the LoRaWAN standards. The end-device attempts to retransmit the same data for confirmed frames and in the event that no ACK is received. In addition to using a different frequency channel, this retransmission may also use a different data rate—ideally one that is lower—than the first one.

The following guidelines should be followed by the DR that is used. The same DR is used for the first and second transmission attempts of a confirmed message; a lower data rate (or DRo if the previous DR was used) is used for the third and fourth transmission attempts; and so on, until the eighth transmission attempt. The MAC layer is supposed to transmit an error code to the upper layer following 8 transmission attempts of the same confirmed message without receiving an ACK.

As seen in Fig.I.12, each retransmission begins after an *ACK* timeout (*ACK_TIMEOUT*) period that begins at the start of the last second receive window. As a result, the retransmission begins between *RXDelay*2 + 1 and *RXDelay*2 + 3 seconds after the original transmission ends. The previous *DR* utilized is used for any future transmission.

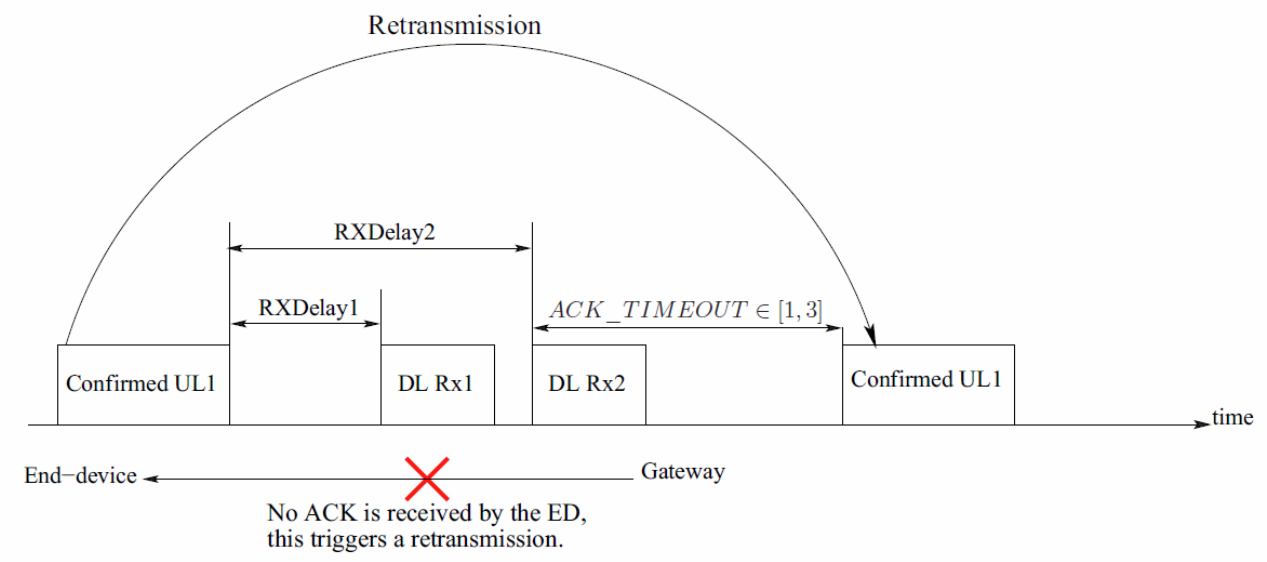


Fig. I.12 – *LoRaWAN Retransmission Procedure.*

I.6 Open Issues

Many Internet of Things applications can benefit from the LoraWAN protocol. However, there are certain restrictions with LoRaWAN that make it unsuitable for usage in any IoT application. Several studies looked into the LoRaWAN network's capacity issue by figuring out the causes of various restrictions.

The primary constraints that affect the network's scalability are covered below:

• **Compatibility:** One of the main issues facing LoRaWAN is communication between the heterogeneous devices, which are made by various vendors. Therefore, maintaining device interoperability across the network is a key objective in the Internet of Things area.

• **Battery Life:** Another issue that LoRa devices must deal with is energy consumption because the majority of them (class A and B) run on batteries, making energy efficiency crucial. Not only may replacing the battery be expensive because batteries can occasionally cost more than the device itself, but it can also be difficult if the device is difficult to access. To extend the battery's duration, the solution must be effective.

• **Complexity:** This is another challenge that LoRaWAN needs to overcome. Adding a new communication feature to the LoRaWAN architecture is one challenging and time-consuming procedure. So, an abstraction layer must be present in the LoRaWAN architecture to make it easier to add new functionality.

• **Security:** LoRaWAN has some weaknesses in security that leave it open to various attacks, including bit-flipping and replay attacks. It is crucial to test LoRaWAN's security features, such as encryption, authentication, and integrity, to guarantee data privacy and defense against intrusions. Therefore, creating security solutions that are lightweight is essential for LoRaWAN small devices, which have limited power, processing, and storage capabilities. But reaching a high level of security means using more processing power and time, which adds to the cost in terms of computation and delay.

• **Privacy protection:** IoT devices may have private data that needs to be secured. Efficient data security techniques must be implemented, though, as several dangers continue to compromise private IoT data.

• **Coverage and Range:** LoRaWAN is designed to provide long-range communication, but its performance heavily depends on the specific deployment environment. Testing the coverage and range requires careful consideration of factors like signal interference, obstructions, and topography that can impact signal propagation.

• **Interference and Noise:** Since LoRaWAN uses unlicensed *ISM* bands for operation, it must share the spectrum with other systems and devices. LoRaWAN connection dependability and overall performance might be impacted by interference and noise from other wireless devices.

• **Scalability:** Numerous devices connected to LoRaWAN are commonly part of Internet of Things deployments. It is critical to test the network's scalability to support a large number of devices transmitting data simultaneously.

• **Latency and Throughput:** Compared to other wireless technologies, LoRaWAN may have higher latency and lower throughput as it is designed for low-power, long-range communication. For Internet of Things applications that have particular latency and data rate needs, evaluating and controlling these trade-offs is essential.

• **Network Server Performance:** Message routing, device registrations, and security management are all managed by the LoRaWAN network server. It is essential to assess the manner in which the network server performs and how successfully it can manage escalating device loads.

For an IoT deployment to be successful and long-lasting, these problems and difficulties with LoRaWAN testing and deployment are crucial. They facilitate more informed decision-making, stronger security, better network performance, and higher levels of user pleasure all around. Comprehensive testing approaches that incorporate simulation, validation, and real-world testing are required to overcome these obstacles. Furthermore, cooperation between network providers, hardware producers, and developers of IoT applications might result in more dependable and durable LoRaWAN installations in the Internet of Things.

Current testing and performance evaluation methodologies of LoRa and LoRaWAN in IoT applications: Classification, issues, and future directives

I.7 Conclusion

In the first section of this chapter, we discussed the reasons for the popularity of LPWAN, explained LoRa, NB-IoT, and Sigfox, and compared these LPWAN networks based on IoT characteristics like power consumption, deployment costs, payload capacity, quality of service, network coverage, and flexibility. We conclude that NB-IoT is appropriate for applications that need very low latency and good quality of service, whereas LoRa and Sigfox are appropriate for applications that need long coverage, low deployment costs, and extended battery lifetimes.

The LoRa physical layer was then thoroughly described, including the sub-GHz ISM bands, the *CSS* modulation, and the physical layer's parameters.

We introduced the LoRaWAN architecture and present the different network components, including end devices, gateways, network servers, application servers. We covered all three of the classes that LoRaWAN defines. Lastly, a few unresolved problems with the new network were covered.

Impact of Mobility on Lorawan Performance.

II

Contents

This chapter consists of two parts: In the first part, we will present an in-depth analysis of the performance evaluation of LoRaWAN in the case of mobility. To do this, we use the $NS3$ simulator to examine various types of scenarios and performance metrics, focused on three mobility models, such as Gauss Markov model, the constant position model, and the random waypoint model. In Part two, we have carried out a number of experiments with the Lora end device $CubeCellHTCC - AB01$ model in various scenarios by analyzing the $RSSI$ (Received Signal Strength Indicator) level in both urban and rural areas using a large number of trajectories in order to validate the simulation results.

II.1 Introduction

Any network needs to take mobility into account. Wireless Sensors Networks (WSNs) allow nodes to move freely in any direction and at any moment. High mobility can cause damage to network architecture and may also be the cause of strange network communication behavior. As such, mobility is a feature of many IoT applications and solutions. In summary, the mobility features of LoRaWAN can be helpful in many different fields, such as smart cities, asset tracking, agriculture, healthcare, and industrial applications. These qualities enable dependable long-range communications in places where conventional cellular networks might not be present. Packet size, latency, energy consumption, and Packet Delivery Ratio (PDR) of LoRaWAN networks are affected by the growing number of End Devices (ED) and node mobility models.

A mathematical model that depicts how nodes move throughout a network is usually referred to as a mobility model in the context of LoRaWAN. A LoRaWAN network's behavior can be simulated, and its performance may be evaluated under various conditions, using a mobility model. In the context of LoRaWAN, a variety of mobility models, such as Gauss-Markov, Random Walk, and Random Waypoint models, can be employed. These models can be used to represent different scenarios, but they differ in how they assume nodes to move. For instance, the Gauss-Markov model postulates that nodes move in accordance with a stochastic process that is influenced by previous placements, whereas the random waypoint model indicates that nodes move arbitrarily within a given area. These models can replicate many scenarios, e.g., nodes moving randomly inside a specific area or nodes moving along a predefined path.

The mobility level of a node is determined by computing the difference between its previous and current positions.

The following Equation II.1 is used to find out the level of mobility [43].

$$ML = \sqrt{(x_{\text{new}} - x_{\text{curr}})^2 + (y_{\text{new}} - y_{\text{curr}})^2} \tag{II.1}$$

where (x_{new} , y_{new}) are the coordinates of the sensor node at the new position and (x_{curr} , y_{curr}) are the coordinates points of nodes at the last calculated position. There is an inverse relationship between a node's mobility and chances for becoming a cluster head; therefore a node having high mobility level has fewer chances to be selected as a cluster head.

Studying its performance in terms of scalability capacity and transmission quality is essential for this. Various authors have improved the ADR rate adaption technique in order to address the problem of assessing the scalability and performance of LoRa for static nodes. Furthermore, the mobility scenario is not taken into account by the ADR debit adaptation technique that is currently in use. To the best of our knowledge, the literature has not examined or discussed how mobility affects Lorawan performance.

II.2 Mobility Patterns

Mobility models, which can be loosely classified into four subclasses as illustrated in Fig.II.1, describe the movement patterns of mobile parts. Models of random mobility: Random Direction (RD), Random Waypoint (RWP), Random Walk (RW), models with geographical reliance (Probabilistic Random Walk), models with temporal dependency (Gauss Markov (GM), Semi-Markov Smoot), and models with geographic constraint (Pathway). The three potential mobility patterns used in this study to explain the node's movement are the RWP, GM, and Constant Position (CP) models.

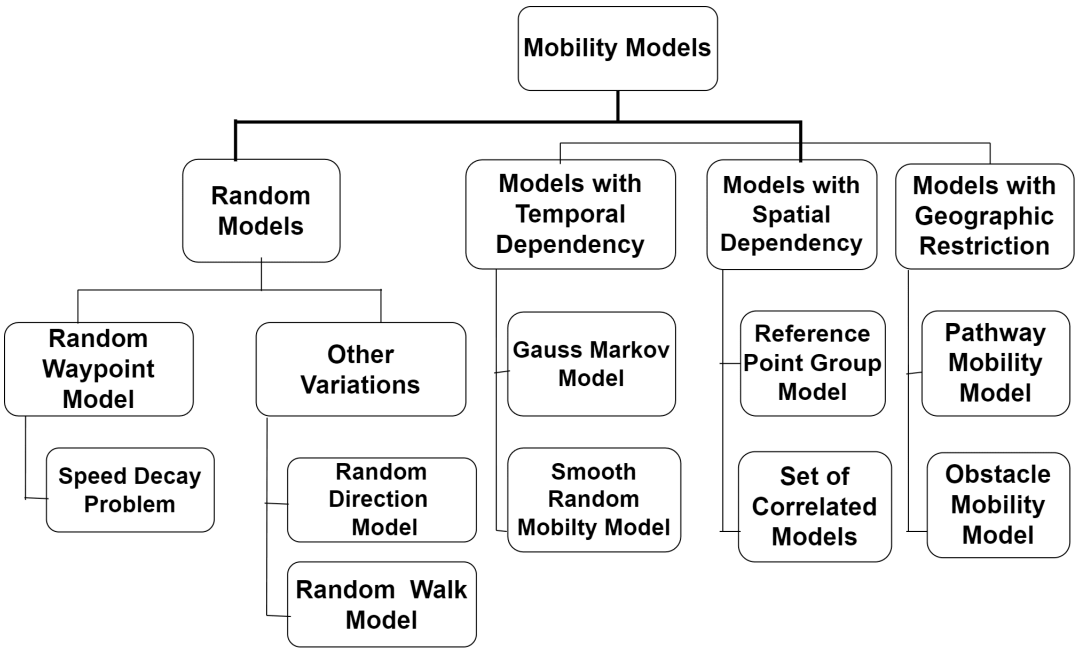


Fig. II.1 – *Mobility Models.*

II.2.1 Random Waypoint Mobility Model (RWP)

Johnson and Maltz were the ones who first introduced the Random Waypoint mobility concept [44]. Because of its simplicity, it is the most frequently utilized for mobile ad hoc networks [45]. The Random Waypoint mobility model assigns each node a speed that is randomly distributed between $[Vmin, Vmax]$ and a random destination inside the simulation zone. The node comes to a stop at the destination for the amount of time specified by the t pause parameter. Continual movement results if t pause = 0. Upon expiration of this time period, the node once more selects and moves toward a different random location within the simulation region. Until the simulation is over, the entire process is repeated numerous times [46], as shown in Fig.II.2.

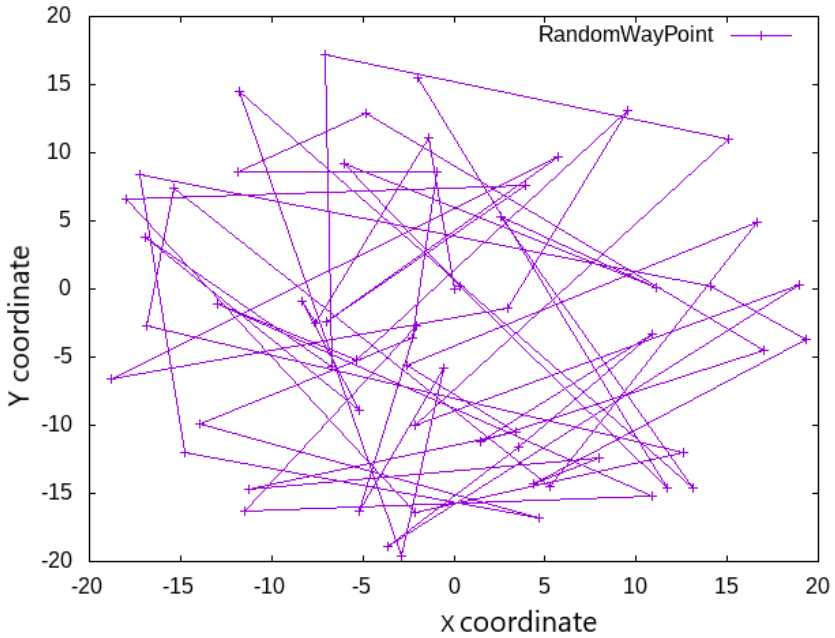


Fig. II.2 – *Trajectory of End Device under the random waypoint (RWP) model.*

II.2.2 Gauss-Markov Mobility Model (GM)

The Gauss-Markov mobility model was originally used for network protocols proposed for wireless Personal Communications Systems [44]. The model works on a timeslot basis and one tuning parameter [45] allows the Gauss-Markov mobility model to adjust to various amounts of unpredictability. Each node is first given a current speed and direction. Nodes move by updating their respective speeds and directions at set intervals of time n. Specifically, This is achieved by determine the speed and direction at the nth instance based on the speed and direction at the $(n-1)$st instance and a random variable, this cycle repeats through the duration of the simulation. The new speed and direction parameters are calculated using the following equations:

$$v_n = \alpha v_{n-1} + (1-\alpha)\bar{v} + \sqrt{1-\alpha^2} v_{x_{n-1}} \tag{II.2}$$

$$d_n = \alpha d_{n-1} + (1-\alpha)\bar{d} + \sqrt{1-\alpha^2} d_{x_{n-1}} \tag{II.3}$$

where:

- v_n and d_n are the new speed and direction of the node at time interval n.
- $\bar{v}$ and $\bar{d}$ are constants representing the mean value of speed and direction as n $\to \infty$.
- $v_{x_{n-1}}$ and $d_{x_{n-1}}$ are random variables from a Gaussian distribution that give some randomness to the new speed and direction parameters.
- α is the tuning parameter, where $0 \leq \alpha \leq 1$.

➢ The tuning parameter α controls degree of randomness in mobility pattern.

• When α is close to 0, mobility pattern is more random and less correlated with previous state.

• When α is close to 1, mobility pattern is more predictable and more correlated with previous state.

Therefore, by varying α, model can adapt to different levels of randomness and correlation in mobility scenarios.

At each time interval, the next location is calculated based on the current location, speed and direction of movement. Specifically, at time interval n, a node's position is given by the equations (II.4) and (II.5) :

$$x_n = x_{n-1} + v_{n-1} \cos d_{n-1} \tag{II.4}$$

$$y_n = y_{n-1} + v_{n-1} \sin d_{n-1} \tag{II.5}$$

where $(x_n\ ,\ y_n)$ and $(x_{n-1}\ ,\ y_{n-1})$ are the x and y coordinates of the (Mobile Node) MN's position at the nth and $(n-1)$st time intervals, respectively, and (v_{n-1}) , (d_{n-1}) are the speed and direction of the MN, respectively, at the $(n-1)$st time interval.

❑ **Gauss Markov Variables:**

➤**Alpha α:** The variable α is the tuning parameter for GM. Setting α to a value between 0 and 1 allows us to adjust the degree of the model randomness.

- **Special Case:** $\alpha = 1$**:**

  When α=1, the model will be predictable and lose all of its randomness, and the new direction and velocity will be the same as the previous direction and velocity. Thus, when α=1, the node moves in a straight line:

$$v_n = v_{n-1} \tag{II.6}$$
$$d_n = d_{n-1} \tag{II.7}$$

Fig.II.3 shows an example of a GM trajectory in a 2D area. Linear motion is obtained by setting α = 1.

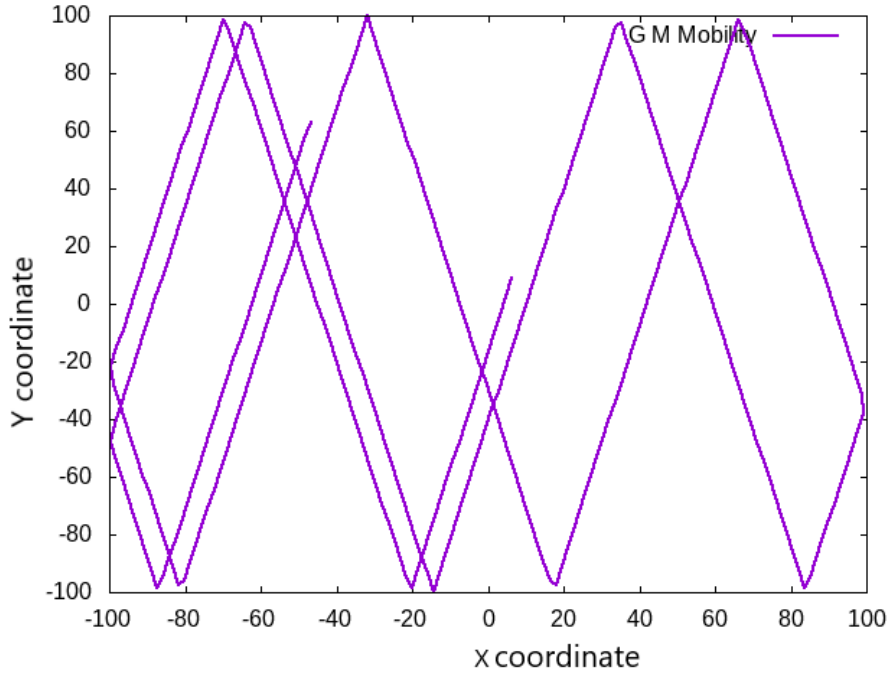


Fig. II.3 – *Gauss Markov Model, Alpha = 1.*

- **Special Case:** $\alpha = 0$ **:**

  On the other hand, when α=0, the model will become memoryless, and the new direction and velocity will depend completely on the mean and standard deviation values of the direction and velocity and the Gaussian random variables.

$$v_n = \bar{v} + v_{x_{n-1}} \tag{II.8}$$
$$d_n = \bar{d} + d_{x_{n-1}} \tag{II.9}$$

Various levels of memory and randomness can be accommodated by setting α between zero and one. Apart from α, the Gauss-Markov mobility model's dynamics are also highly dependent on other factors, such as the time step, the choice of average speed and direction, and the mean and standard deviation of the Gaussian random variables. For instance, a substantially different movement pattern results from selecting a standard deviation on the Gaussian distribution regulating the direction that is significantly bigger than the average direction, as opposed to if the values of the standard deviation and average direction are identical.

Fig.II.4 shows an example of a GM trajectory. A Brownian motion is obtained by setting Alpha α = 0.5.

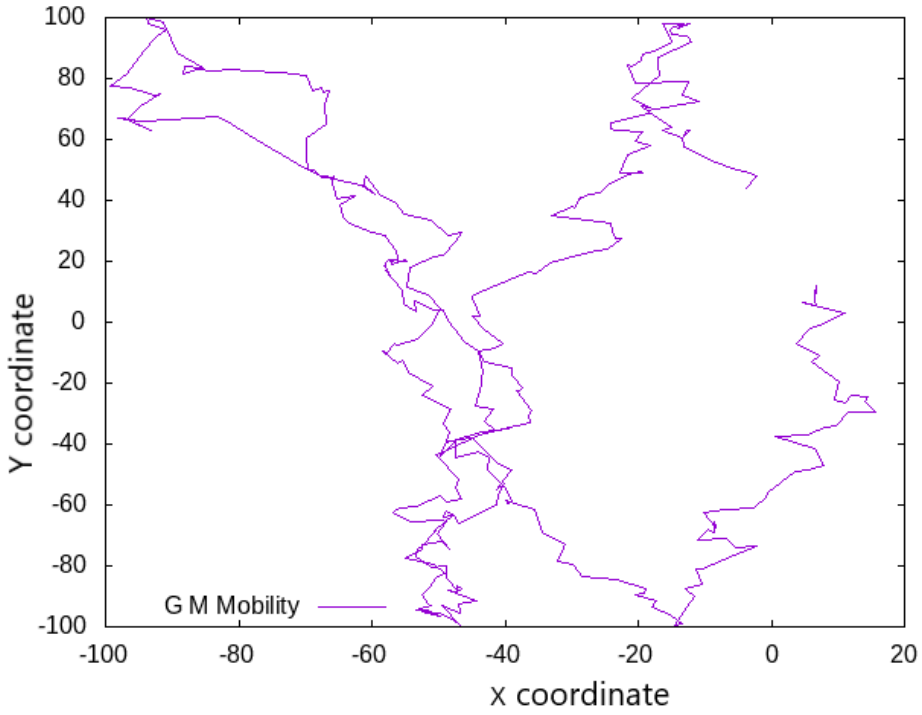


Fig. II.4 – *Gauss Markov Model, Alpha = 0.5.*

II.2.3 Constant Position Mobility Model (CP)

The constant position Mobility model helps us in simulating an environment where there is no movement of the nodes. The initial position of the nodes will be set say (x, y) and they remain in the same position unless explicitly changed [47]. This can be applied to vehicles in a parking lot or in an environment of a heavy traffic jam where the vehicles are in the same position for a longer time.

II.3 Energy Framework

Energy usage is a critical concern for wireless devices, and researchers in wireless networking often need to analyze the energy consumption of nodes or the overall network during network simulations using NS-3 [40]. To facilitate this, NS-3 must include support for energy consumption modeling. Moreover, with the growing viability of concepts like fuel cells and energy scavenging for low-power wireless devices, it is crucial to enable modeling of various energy sources in NS-3 in order to integrate the effects of these new technologies into simulations [48]. Energy harvesting, energy sources, and energy consumption are all modeled using the NS-3 Energy Framework[49].

• **Energy Consumption (EC):** The energy consumption (EC) determines the amount of energy consumed in joule by of an End Node. It depends on SF, BW, CR, and TP. The correct choice of transmission parameters leads to decrease the transmit energy consumption and is defined as :

$$EC = \sum_{i \in \text{ packets}} \text{SupplyCurrent }_i * t_i(\text{ Transmitting }) \tag{II.10}$$

The factor $SupplyCurrent_i$ refers to the power setting used by the end node to transmit packet i, while $t_i(Transmitting)$ refers to the time used by the radio to transmit packet i.

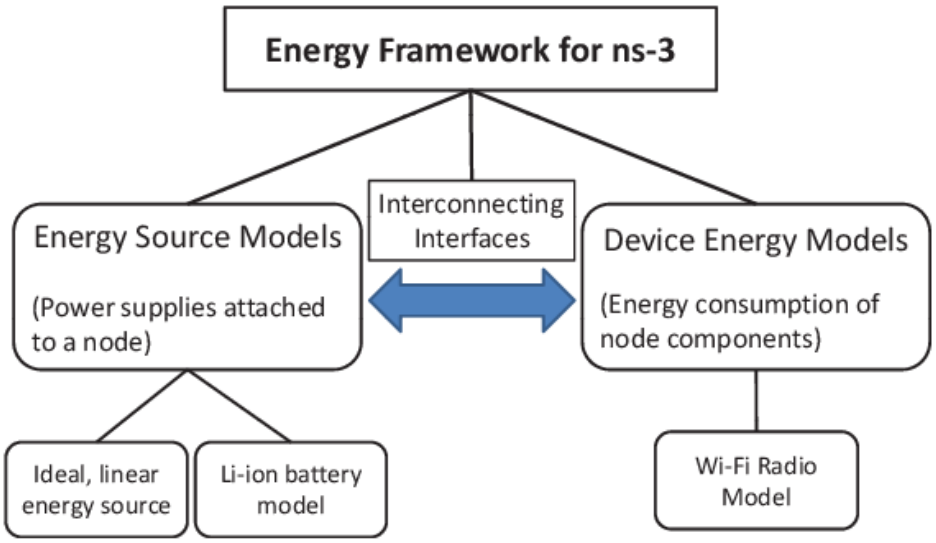


Fig. II.5 – *NS-3 Energy Framework Structure Proposal, which includes energy sources, device energy models, and their interfaces.*

II.3.1 Energy Source:

The power source on each node is represented by the Energy Source. Each energy source on a node can be connected to several device energy models, and a node may have one or more energy sources. When an energy source is connected to an energy model, it is assumed that the matching device is powered by the source. Energy supply for the devices on the node is the Energy Source's fundamental function. The devices on the node are alerted when the Energy Source is totally depleted so that they can all respond accordingly. Additionally, each node has access to the Energy Source Objects for data like the amount of energy left or the energy fraction (battery level). As a result, ns-3 may now implement energy-aware protocols.

The Energy Source needs to be able to capture the properties of these power supplies in order to simulate a wide range of power sources, including batteries. Practical batteries have the following 2 crucial traits or results:

1. ***Rate Capacity Effect:*** Reduction in battery life when the current demand exceeds the battery's rated value.

2. ***Recovery Effect:*** The battery's lifespan is extended when it alternates between discharge and idle conditions.

   The Energy Source calculates energy consumption using current draw from all devices on the same node in order to account for the Rate Capacity Effect. Additionally, the Energy Source can be reenergized by connecting several Energy Harvesters to it.

To determine the overall current drain and, consequently, the energy consumption, the Energy Source polls all of the devices and energy harvesters on the same node on a regular basis. When a device's state changes, the Energy Source is informed by the device's related Device Energy Model, and a new total current draw is computed. In a similar manner, the linked Energy Source is updated in response to each Energy Harvester update. The Energy Source basic class maintains a list of the energy harvesters (Energy Harvester objects) and devices (Device Energy Model objects) that are powered by the specific Energy Source. All devices on this list will receive a notification from the Energy Source when the energy is fully depleted. After that, each device can respond to this situation on its own, depending on the planned course of action in the event of a power outage.

II.3.2 Device Energy Model:

The energy consumption model of a device that is put on the node is called the Device Energy Model. The model is intended to be state-based, with the assumption that every device has a number of states, each of which is linked to a power consumption figure. The matching Device Energy Model will alert the Energy Source to the device's new current draw whenever its status changes. After that, the Energy Source will update the remaining energy and determine the new total current draw.

It is possible to apply the Device Energy Model to devices with an infinite number of states. For instance, in an electric car, the motor's current demand is dictated by its speed. It is impossible to establish a set of discrete states of operation for the vehicle because its speed can assume continuous values within a specific range. However, the same set of Device Energy Model APIs can still be utilized by directly translating the speed value into current.

II.3.3 Energy Harvester:

The components of the energy harvester are those that capture environmental energy and replenish the energy source to which it is attached. The energy harvester encompasses the full integration of the environment (such as solar radiation) and the energy gathering device (such as a solar panel). This means that in order to execute an energy harvester, it is necessary to jointly simulate the energy contribution of the environment and the additional energy requirements of the energy collecting device, such as the conversion efficiency and the internal power consumption of the device.

II.4 Signal characteristics

There are differences between wired and wireless communication: the signal strength will decrease when the signal passing through walls and other objects. If the distance between the sender and the receiver is too great the signal will also decrease to an unreadable signal or not reach the receiver at all. One other parameter impacting the signal is interference from other sources, mainly sources sending in the same frequency band. A signal can also be reflected on the ground or on objects, causing the same signal to reach the receiver multiple times, each instance slightly offset in time from the other. The phenomenon is called multipath propagation and will cause the receiver to register the additional signals as noise, causing the signal to be less clear [50].

II.4.1 Signal to Noise Ratio (SNR)

The power of the signal divided by the power of the noise is known as the signal-to-noise ratio, or SNR. The logarithmic decibel scale is used to indicate the ratio. P_{signal} is the power of the signal and P_{noise} is the measured power of the noise. Note that (as equation (II.11) implies) the SNR is a unitless ratio, although often converted to dB.

$$SNR_{dB} = 10 * log_{10}(\frac{P_{signal}}{P_{noise}}) \tag{II.11}$$

A ratio higher than 1: (1> 0 dB) indicates that the signal is stronger as the noise. A higher positive SNR therefore indicates a better reception of a signal while a SNR of less than 0 dB, a ratio lower than 1. It is also the case that a larger SNR means a lower probability of error, and thus a signal with more power is more likely to be received and decoded correctly.

II.4.2 Received Signal Strength Indicator (RSSI)

In the field of telecommunications, Received Signal Strength Indicator ($RSSI$), is a measurement of the relative signal strength of wireless network transmissions or, alternatively, a measure of the radio receiver's power level. Stronger received signal is indicated by a higher $RSSI$ value. $RSSI$ is measured in the decibel-milliwatts (dBm). The interference from sources other than the part sending is also included in the received $RSSI$. $RSSI$ is often represented in negative dBm, which means that a value closer to 0 indicate a better signal. A $RSSI$ of -60 dBm can be considered rather good while -100dBm is considered less good.

II.4.3 Packet Delivery Ratio (PDR)

This can be calculated by dividing the total number of packets transmitted by the end nodes by the total number of packets received by the network server. The PDR can be computed per node or for the whole network. It is one of the well-known performance metrics in the sensor networks literature. For the entire network, this can be computed as shown by Equation:

$$PDR = \frac{\sum \text{ Number of packet received}}{\sum \text{ Number of packet sent}} \tag{II.12}$$

II.4.4 Delay (Second)

The Delay is measured from the moment the packet leaves the source application to the moment the same packet arrives at the destination application:

$$Delay = \frac{\sum(\text{Time Packet Received} - \text{Time Packet Transmitted})}{\text{Number of Packets Received}} \tag{II.13}$$

II.5 Similar Works

Many researchers conduct interesting studies and papers on improving the performances of the Internet of Mobile Things by employing various techniques.

Authors [51] proposed Resource Management Adaptive Data Rate for Mobile Application (RM-ADR). ADRs on the NS and ED sides for transmit power management and spreading factor. Using retransmission information, RM-ADR, which is implemented at the ED side, assigns SF and TP to the ED after counting the number of transmissions from each ED that are sent to NS in the LoRa frame header. However, the suggested RM-ADR at the NS side allocated SF and TP parameters based on the received power after extracting the number of transmission details from the LoRa frame header and comparing them to a packet transmission threshold. The suggested RM-ADR performed better in terms of convergence time, PDR, and energy usage when compared to the most advanced ADR schemes.

Authors [52] putting forth two ADR proposals that will be managed by Network Server NS, the Gaussian filter-based ADR (G-ADR) and the Exponential Moving Average-based ADR (EMA-ADR). Both of these methods function as a low-pass filter, with the aim of resisting sudden changes in the signal-to-noise ratio of received packets at the NS. their proposed approaches are geared towards the allocation of the most suitable Spread Factor SF and Transmit Power TP to both mobile and static EDs, with the ultimate goal of reducing convergence time in the confirmed mode of LoRaWAN. Similar to [53], the authors employed simulation tools in order to show that end device movement leads to erroneous configuration commands for determining an end device's spreading factor and transmission power, the authors used simulation tools.

In [54] proposed an Improved ADR (IADR) to address the initial SF 12 problem in ADR. IADR aimed to assign all SFs concerning the received signal strength between the ED and GW

during the network deployment. The IADR method [54] when compared to the ADR, improved the PDR. The performance results of the proposed scheme with the ADR show an improved packet delivery ratio. The authors also highlight the convergence time issue in their work. However, it failed to present a solution and resolve the convergence time issue. The same authors [55] proposed Hybrid Adaptive Data Rate (HADR) for IoT that uses LoRa's ADR mechanism to improve the performance in cases where there are both fixed and moving users, using the same MAC protocol. HADR constitutes a hybrid model in terms of parameter selection. This approach led to increased packet success rate compared to other approaches, such as Adaptive Data Rate (ADR) and blind Adaptive Data Rate (BADR). Recently[56], the distance between the ED's former and present positions at the NS is used by the proposed Mobility ADR (M-ADR) to determine the ED status (i.e., whether it is static or movable). They employed the Kalman Filter to estimate the Signal-to-Noise Ratio (SNR) to precisely determine SF, TP, or both if the ED status was found to be mobile because these parameters are mostly dependent on SNR. The suggested M-ADR further determines the ideal SF and TP configuration after the Kalman Filter determines the system's current estimate.

They [57] describe and evaluate a new and effective method for determining the level of mobility of end devices based on machine learning techniques and specifically on the support vector machine supervised learning method. The proposed method does not rely on the location capability of LoRaWAN networks; instead, it relies only on data always available at the LoRaWAN network server. Moreover, the performance of this method in a real LoRaWAN network is assessed; the results give clear evidence of the effectiveness and reliability of the proposed machine learning approach.

The Linear Regression-ADR (LR-ADR) mechanism is proposed by [58] for the Network Server side in order to anticipate the SNR of the next transmission and smooth the Signal to Noise Ratio (SNR) estimations per gateway. Second, they suggest the Linear Regression plus ADR (LR+ADR) mechanism, which is an adaptive technique that helps the end-device side reconnect to the Network Server more quickly. They compared the results with those of four alternative solutions—ADR, ADR+, EMA-ADR, and G-ADR—and used simulation modeling to assess the effectiveness of the implementation. The first strategy (LR-ADR) outperforms the best competition, according to the data, and the second approach (LR+ADR) improves the PDR even more while maintaining low levels of Energy Consumption per Packet Delivered (ECPD). Additionally, SF and TP values were found to be comparable to the normal ADR when the average of the previous 20 SNR values was calculated.

The resource assignment problem in static and mobile IoT systems is addressed by this article [38], which suggests a unique proactive solution called "artificial intelligence-empowered resource allocation" (AI-ERA). There are two modes in the AI-ERA approach: offline and online modes. Initially, a dataset created at ns-3 in the offline mode is used to train a deep neural network (DNN) model. Second, the suggested AI-ERA method proactively assigns an efficient SF for the end device prior to each uplink packet transmission by using the pretrained DNN model in the online mode. When compared to the standard LoRaWAN ADR, the AI-ERA's proactive behavior increased the packet success ratio by an average of 32% in static settings and 28% in mobility scenarios.

To improve the convergence period of the usual ADR, [59] presented an enhanced ADR (EADR) under a static scenario at the NS. M was set to 5 in the EADR rather than 20 UL packets. Upon reaching M received UL packets, the EADR calculates the PSR and compares it to an 80% threshold. Next, it determines the SNR values of M packets' standard deviation. To determine an appropriate SF and TP, the standard ADR was started if the standard deviation result was less than 2.5 dB. In comparison to the standard ADR, their simulation findings showed that EADR improved the PSR and convergence period. However, there is a lengthy convergence period because the EADR at the NS side always waits for M packets following a DR change.

In order to reduce packet collisions in the LoRaWAN network, another ADR was suggested

in [60]. Three steps make up their suggested ADR: 1) documenting the time at which successive transmissions arrive at the NS, 2) grouping EDs according to the same SF, and 3) reducing the collision by assigning the EDs engaged in the collision the appropriate SF, TP, and UL transmission time. By lowering the number of retransmissions, their simulated results reduced the convergence period and energy usage while improving the PSR.

The authors in [61] suggested a new-dynamic ADR (ND-ADR) to dynamically modify the value of M (notice that the value of M is set to 20 in the usual LoRaWAN ADR). The value of M in the ND-ADR is originally set to 3. Instead of selecting the highest SNR value among M, ND-ADR then combines the RSSI and the average of the SNR (SNRavg) values of M received packets for the demodulation. The value of M is raised when SNRavg exceeds or falls below a specific threshold established for every SF. When compared to standard ADR, their OPNET simulation findings demonstrated better throughput, latency, and energy consumption.

Prediction of LoRaWAN behavior using ML was reported in [62] .First, unsupervised learning was utilized for profiling EDs (i.e., clustering) through a well-known K-means method. The profiling method aimed to group the EDs with similar transmission characteristics (e.g., the same SF and packet size). Then, the decision tree classifier and long short-term memory (LSTM) models were used to predict traffic patterns.

In [63], a regression method utilizing an LSTM based on an extended Kalman filter (EKF) was presented to anticipate collisions. Based on a pretrained traditional LSTM model, the authors employed the LSTM-EKF model as the framework. Predicting future collisions was the main goal, and the LoRaSim simulator was used to create the dataset. According to the simulation results, the LSTM-EKF model outperformed the traditional LSTM in terms of future collision prediction. To reduce collisions by assigning an appropriate SF to EDs, no proactive approach has been put forth.

Benkahla et al. [64] developed an Enhanced ADR (E-ADR). E-ADR is an improved version of the ADR that supports rate adaption for mobile end devices but also works for static end devices. The algorithm is solely controlled by the network server and therefore, unlike in the ADR, the network server has the capability to increase and decrease transmission parameters accordingly. The algorithm adapts data rates for end devices based on their locations and predetermined trajectory. The E-ADR was designed to minimize airtimes, energy consumptions, and packet error rates in mobile end devices.

II.6 Choice of Simulator

The majority of network simulation software on the market depends on Discrete Event-based Simulation (*DES*), in which events are triggered by nodes inside the simulated network. The simulator keeps an events queue that is arranged by the event's scheduled execution time. Events in the queue are processed one after the other to carry out the simulation itself. Over the past 20 years, computer network simulation has started to use the *DES* technique. After the initial implementations, *NS*-2 emerged as the standard for network simulation and has remained so ever since. The implementation of numerous network component models in $NS-2$ is the reason behind this.

However, a significant limitation of $NS-2$ [65] is its restricted scalability concerning memory utilization and simulation execution time. This is a significant issue for recent developments in computer network research, like mesh designs, peer-to-peer networks, and wireless sensor networks, which call for the simulation of extremely large networks. Several enhancements, like parallelism, have been suggested for $NS-2$ to address these issues. But a significant overhaul of $NS-2$ is now underway. Enhancing simulation performance is one of the primary objectives for development of $NS-3$, its successor.

Universities and industry presently use over a dozen network simulators in addition to $NS-2$ [66]. The most well-known simulators include commercial simulators like *OpNet*, spe-

cialized simulators like the *TOSSIM* wireless sensor network simulator, royalty-free simulators like $OMNeT++$, *JIST*, and $NS-3$. Moreover, $NS-3$ unifies code and architecture ideas for a scalable simulator. The compromise made in these design choices was the loss of $NS-2$ compatibility.

A comparison of many simulators was done in [67], which demonstrated the effectiveness of $NS-3$, $OMNeT++$, and *JiST* in large-scale network simulations. The properties of the simulators have been comprehensively and clearly illustrated in Table II.1.

Table II.1 – *Comparison of network simulation tools.*

Features	NS2	NS3	JSIM	OMnet ++	OPNET	LoRaSim
Langage	C++ / OTcL	C++ , Pyton	Java , Tcl	C++	C , C++	Pyton
GUI	Limited	Limited	Good	Good	Excellent	Only Plot
Platform	Linux,MacOS Windows	Linux,MacOS Windows	Linux,Mac Windows	Linux,Mac Windows	Linux Windows	Linux,Mac Windows
Scalability	Small	Large	Small	Large	Medium	Small
License	Open Source	Open Source	Open Source	Open Source (Study and research)	Commercial	Open Source
Document	Excellent	Excellent	Poor	Good	Excellent	Poor

Today, $NS-3$ is still evolving and some models are still being developed. So we chose $NS-3$ because of its potential for expansion, which is beginning to materialise in the large and growing amount of source code contributed by the $NS-3$ computing community every day.

The Fig.II.6 shows the number of publications cited in Google Scholar using $NS-3$ over the last 10 years compared to other network simulators.

II.6.1 Network Simulator NS-3

An extension of the well-known network simulator 2, $NS-3$ aims to simulate different communications networks as accurately as feasible. In particular, $NS-3$ is a discrete-event network simulator that was started in the middle of 2006 [49]. It is developed in C++ and has Python bindings that allow it to replicate different communication stack levels with a great degree of flexibility. Additionally, The $NS-3$ simulator supports a wide variety of protocols such as Wi-Fi, LTE, IEEE 802.15.4, SigFox, LoRa, further networks and also implements an IP networking with support for more to be added and expanded upon. $NS-3$ was developed primarily for research and educational use. It has been used as the primary tool for researchers to implement and test a variety of network protocols, models and architectures. It has an event based simulation model and offers realistic implementations of the physical layer for many protocols. We choose $NS-3$ over other network simulators such as Opnet, LoRaSIM or Omnet ++ due to its customization capabilities and extensive documentation.

The LoRaWAN module allows for realistic simulations of LoRa communications. It also allows for customization of the physical parameters and offers LoRa-specific abstractions such as Gateways and Network Servers [68], [69]. All of the work done and presented in this thesis is based on the LoRaWAN module, which was initially shown in [70] and was created in 2015–2016 by researchers at the Università degli Studi di Padova's Department of Information Engineering.

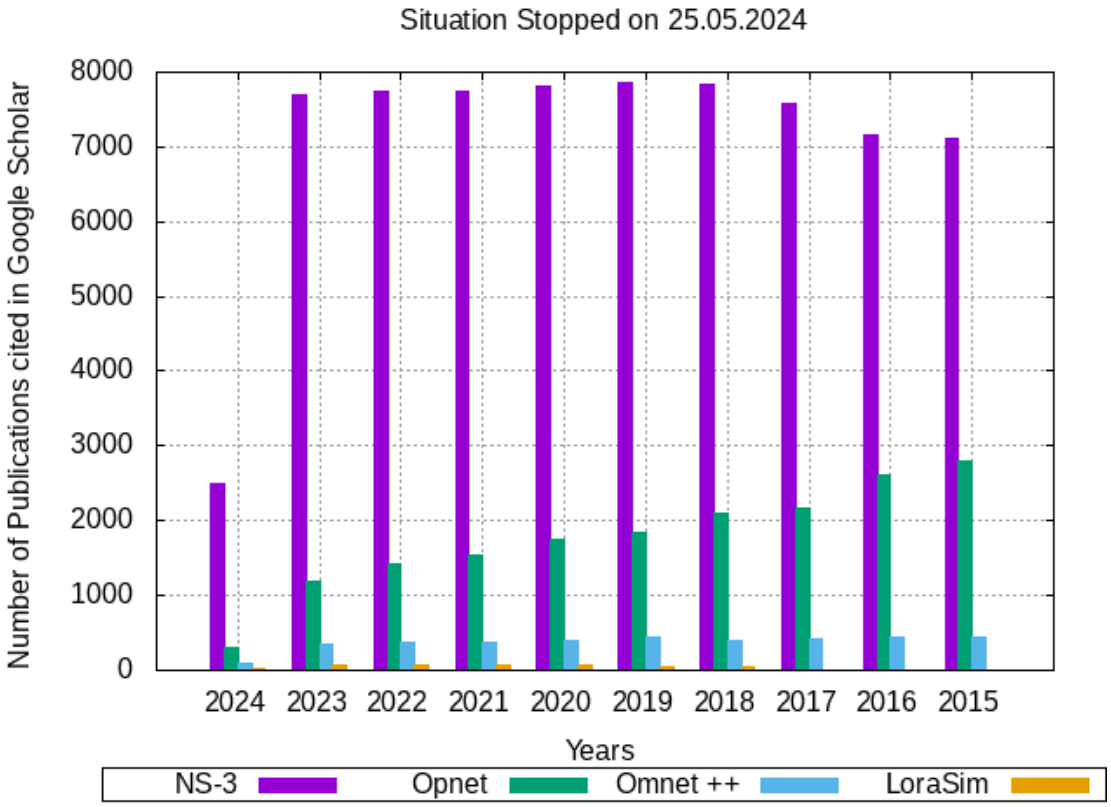


Fig. II.6 – *Network Simulator Popularity by Year.*

II.6.1.1 Installing NS3 under Ubuntu

Once *Ubunto* is installed and before the installation of ns-3 and the LoRa/LoRaWAN module, we make sure that all the operating system packages and prerequisites are upgraded and up to date. We execute the following commands in the terminal to achieve it:

sudo apt-get install gcc g++ python && sudo apt-get install gcc g++ python && sudo apt-get install mercurial python-setuptools git && sudo apt-get install qt5-default && sudo apt-get install python-pygraphviz python-kiwi python-pygoocanvas libgoocanvas-dev ipython && sudo apt-get install openmpi-bin openmpi-common openmpi-doc libopenmpi-dev && sudo apt-get install autoconf cvs bzr unrar && sudo apt-get install gdb valgrind && sudo apt-get install uncrustify && sudo apt-get install doxygen graphviz imagemagick && sudo apt-get install texlive texlive-extra-utils texlive-latex-extra texlive-font-utils texlive-lang-portuguese dvipng && sudo apt-get install python-sphinx dia && sudo apt-get install gsl-bin libgsl2 libgsl-dev && sudo apt-get install flex bison libfl-dev && sudo apt-get install tcpdump && sudo apt-get install sqlite sqlite3 libsqlite3-dev && sudo apt-get install libxml2 libxml2-dev && sudo apt-get install cmake libc6-dev libc6-dev-i386 libclang-dev && sudo pip install cxxfilt && sudo apt-get install libgtk2.0-0 libgtk2.0-dev && sudo apt-get install vtun lxc && sudo apt-get install libboost-signals-dev libboost-filesystem-dev

under the ns-allinone-3.35 folder let's type the command:

./build.py –enable-examples –enable-tests

./waf -d debug –enable-examples –enable-tests configure

./test.py

The following screenshot II.7 shows the successful run and hence proves the good working of the newly installed ns-3 simulator under ubuntu.

Fig. II.7 – *Installing NS3.35.*

II.6.1.2 Add Lorawan Module on NS3

Note that the lorawan module is operational under ns3 from version ns3.32, going to the following page to download the module: https://apps.nsnam.org/app/lorawan/.

Unzip into the ns-3/contrib folder (folder tree is shown in Fig.II.8b), this will allow us to make use of the module to run the simulations.

Once the installation of ns-3 and LoRaWAN module is complete, we now configure and build ns-3 using the following command:

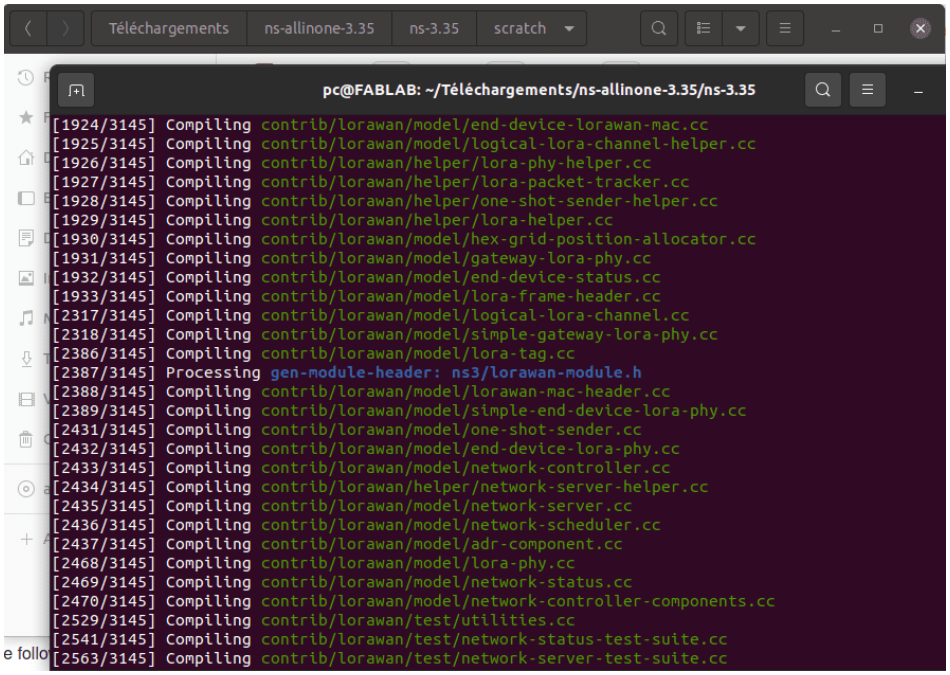
./waf configure –enable-tests –enable-examples

./waf build

This compilation returns results as seen in Fig.II.8a.

II.6.1.3 Details of the implementation of our proposal in NS-3

The idea is to simulate a LoraWan node to which we will add an energy harvesting module (BasicEnergyHarvesterHelper). Script we will use is ***mylorawan-energy.cc***. We will place particular emphasis on the energy part. However, we will describe each block in the example step by step.



(a) (b) *Lorawan Folder Tree.*

Fig. II.8 – *installing LoraWan.*

The first thing we have is creating the nodes. We will create an end device in LoraWan and a gateway. The example is ready to expand its capabilities so that we can put the number of endpoints we want as an argument.

A container of nodes is created with numendDevices, by default equal to one, and they are positioned with a radius equal to 20. We use the createscenario function for this. Next, we create another node container for the nodes that act as gateway or gateway/hub. In this case, also one, and we position it in the center. The createstarcenter function and the createscenario function are at the end of the file. Both return a MobilityHelper which, using containers, we install on endDevices endpoints and hubs.

- NodeContainer endDevices;
- endDevices.Create(numendDevices);
- MobilityHelper scenario = createscenario(radio);
- scenario.Install (endDevices); //put the hub in 0.0 0.0 0.0 center
- NodeContainer hub;
- hub.Create(1);
- MobilityHelper hubposition = createstarcenter();
- hubposition.Install (hub);

Next, we need to put a LoraWan interface on each end device and on each Hub or gateway. To do this, we create a LoraWan channel according to the module documentation: These delay and loss parameters simulate the propagation of a channel using LoRa encoding. We now need to create the Lora interfaces, connected to the channel and we will add them to the nodes already created. Firstly the LoRa interface for end devices using wizards; we

create the physical and MAC layer, indicating that it is an end device via ED and ED_A respectively. Additionally, we do the same with the gateway. Now indicates that this is a gateway.

- LoraPhyHelper phyHelper = LoraPhyHelper ();
- phyHelper.SetChannel (channel);
- LorawanMacHelper macHelper = LorawanMacHelper ();
- LoraHelper helper = LoraHelper ();
- helper.EnablePacketTracking();
- phyHelper.SetDeviceType(LoraPhyHelper::ED);
- macHelper.SetDeviceType(LorawanMacHelper::ED_A);
- NetDeviceContainer endDevicesNetDevices = helper.Install(phyHelper, macHelper, endDevices);

- LoraHelper helperHub = LoraHelper ();
- phyHelper.SetDeviceType (LoraPhyHelper::GW);
- macHelper.SetDeviceType (LorawanMacHelper::GW);
- helperHub.Install (phyHelper, macHelper, hub);
- macHelper.SetSpreadingFactorsUp (endDevices, hub, channel);

Well, the next step is to configure an application, which we will install on the end nodes and which will send a packet every 5 seconds with a size of 12 bytes; We configure the period and package size to install on the end devices. Then we indicate that it starts at second 0 and ends at the start time using a variable, appStopTime which we will use to configure the simulation. The next step we will add to our final device is a power source, a 200 mAh battery and then we will configure the consumption according to a final circuit.

- PeriodicSenderHelper periodicSenderHelper;
- periodicSenderHelper.SetPeriod (Seconds (5));
- periodicSenderHelper.SetPacketSize (12);
- ApplicationContainer appContainer = periodicSenderHelper.Install (endDevices);
- double simulationTime = 3600;
- Time appStopTime = Seconds (simulationTime);

In $NS-3$, a simulation script is launched using the "**waf**" utility which takes care of compilation, linking of libraries and execution of C++ program. This utility allows you to configure the execution of the script by allowing the user to set the values of certain simulation variables from the command line using AddValue:

- CommandLine setupcmd()
- cmd.AddValue("radio", "Radio of the disc where random put the nodes", radio);
- cmd.AddValue("numnodes", "Num. nodes in the grid for simulating",numnodes);
- return cmd;

Subsequently, the mobility model is assigned to each node within a maximum distance defined in the radio command line variable.

Once the simulation is executed, a gnuplot-energy-example.sh file that has been generated with the simulation data, we run it and it generates a graph with the evolution of the remaining energy in the final node during the simulation hour.

II.7 Results and Discussions

This section presents the results of different simulation scenarios by varying the number of end devices, sending periods, number of gateways, packet size, and radius. These results are analyzed and evaluated to study the impact of different mobility models on energy remaining, PDR, and delay in Lora Networks. The simulation goes up to 50, 100, 300, and 500 ED with 1 and 2 GW, the simulation time equals 3600s (1 hour), and a power source: a 200 mAh battery. The node sends data frames to the GW (uplink) with a size of 12 bytes every 5 seconds. The ED positions are randomly assigned around the GW in a radius of 1000, 5000, 8000, 10000, and 15000 meters; a simulator of an energy recuperator is added, which periodically collects a random value. The remaining energy is evaluated, and their effectiveness is compared using the NS-3 simulator; the Lora network is simulated by increasing the number of nodes for the three mobility models.

Table II.2 – *Simulation Parameters.*

Parameter	**Value**	**Unit**
N of Nodes	50, 100, 300 , 500	-
Radius	1000, 5000, 8000, 10000, 15000	Meter
Period	3 , 5 , 7	Second
Packet Size	12, 64, 120, 192, 216	Byte
GateWay (GW)	1 , 2	-
Simulation Time	3600	Second
Mobility Model	ConstantPositionMobilityModel	-
	RandomWayPointMobilityModel vitesse Max = 60 Pause Time = 0	meters/s
	GaussMarkovMobilityModel vitesse Min=20 , Max= 70 Pause Time = 2 , α =1	meters/s
Energy Initial	200	mAh
Batterie PD2032	2664 3.7	Joules Volts
Simulator	NS3 (Version 3.35)	-
Operating System	Ubuntu 24.4 64 bit	-

In the following, the terms "RWP", "GM", and "CP" refer to the "Random Waypoint", "Gauss Markov", and "Constant Position" mobility models respectively.

Fig.II.9 below presents the energy remaining as a function of the simulation time equal to 3600 sec in four (04) scenarios with three mobility models. The impact of the mobility model on

the remaining energy is essential. When 1 ED (end device) sends a packet of size 12 bytes every 05 seconds within a radius of 20m around a single GW (Fig.II.9. (a)), the second case is with two GW (Fig.II.9. (b)), the third case with 500 ED and one GW (Fig.II.9. (c)), and finally, with 500 ED and a packet size of 24 bytes and one GW (Fig.II.9. (d)); energy remaining decreases during the simulation time for all four scenarios. In the first scenario, the GM mobility model with alpha =1, pause time = 2 seconds and a speed varying between 20 and 70 m/s performs better than the two other models with an energy remaining of 2663.9 J. This is because consumption during one hour (simulation time) equals 0.1 J. For the two rest models, RWP and CP, the energy consumption is 0.4J, which equals 9.6 J in 24h, making the GM model 4 times less than the RWP and CP models. In the second scenario, when adding a second The GM mobility model also gives good results compared to the two other models but performs less than the first scenarios. In the third scenario, when increasing the number of ED to 500 nodes. CP and GM have the same value of 2663.6 J, while the worst model is RWP, with a max speed of 60 m/s. In the fourth and last scenario, notice that the GM model has been slightly impacted by the packet size when it is increased by energy remaining of 2663.2 J, while the CP model is better performing with a value of 2663.6 J. The worst model is the RWP of the time pause, which is zero, i.e., maximum mobility. As already seen in the preceding sections, the impact of the mobility model on residual energy, node density, PDR (%), and latency is the primary emphasis of this work.

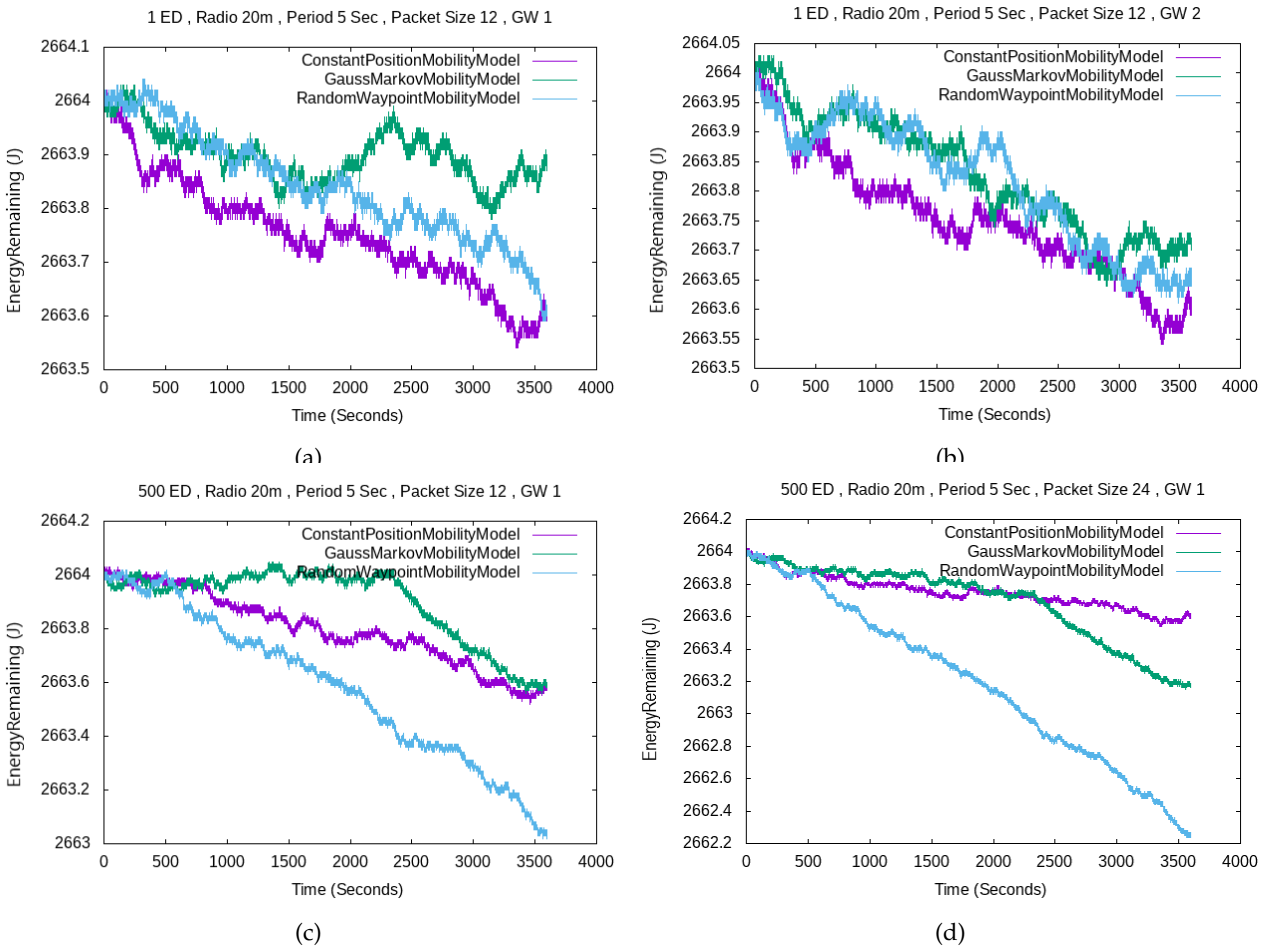
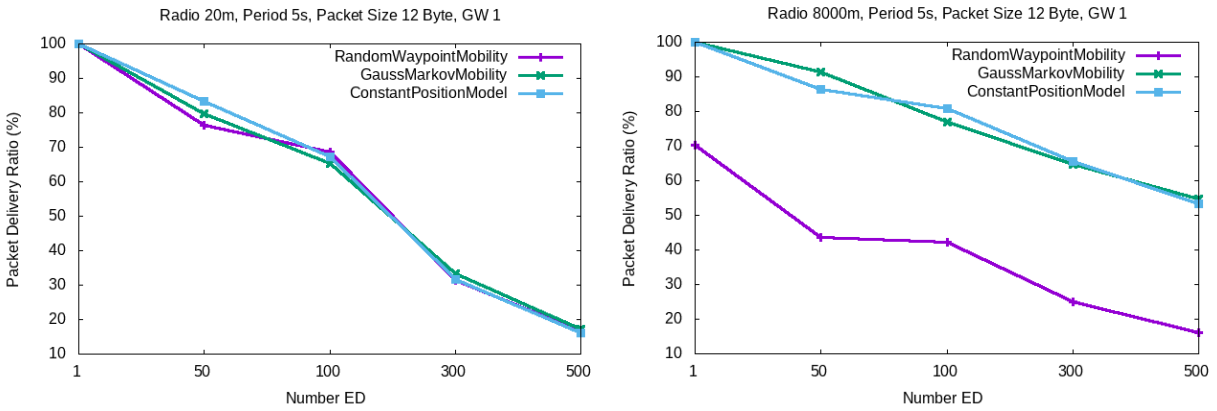


Fig. II.9 – *Impact of Mobility Model On Energy Remaining*

The packet delivery ratio is shown in Fig.II.10 (a) below as a function of the number of nodes. PDR is the ratio of packets delivered to the total number of packets generated. The PDR is inversely correlated to the packet loss rate. A higher packet delivery rate means less packet loss. When there are more nodes in the network, there are more collisions, which causes the

PDR to decrease. At 500 ED in a 20-meter radius around the GW, the graph shows almost the same performances as the three mobility models. The PDR decreases with an increase in the ED by less than 20% since collisions during transmission are less frequent when the number of ED is lower and the collisions increase. They are keeping the same simulation parameters, except the distance increase between ED and GW to 8000 meters, as shown in Fig.II.10 (b). CP and GM provide a PDR greater than 50%, but RWP provides the worst PDR of always less than 20%.

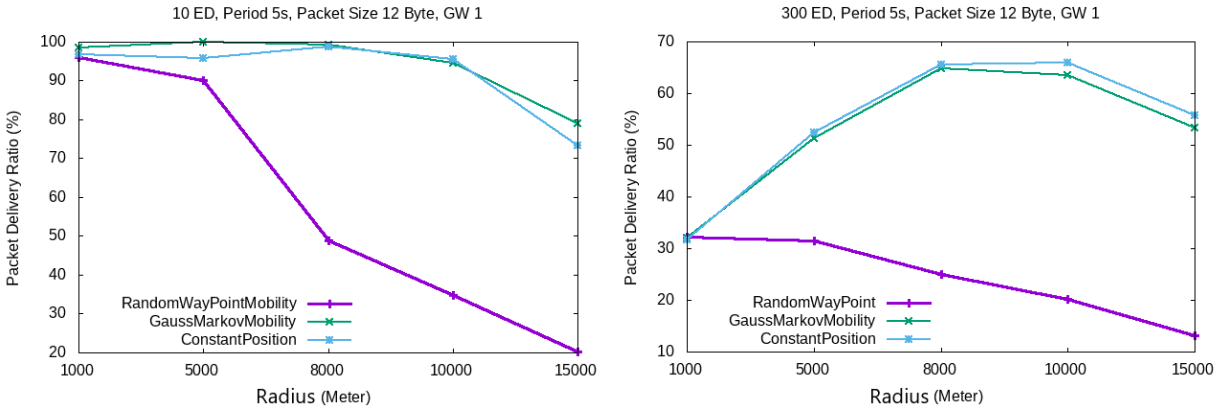


(a) (b)

Fig. II.10 – *PDR Vs Number ED*

Fig.II.11. (a) shows the PDR as a function of radius and follows the same behavior as the number of nodes. When the radius increases, it leads to a decrease in the PDR. With 10 ED. GM marks good results when showing a percentage of 80%, CP slightly less than 70%; unlike the first two, poor results are presented by the RWP, from 8000 m is a decrease of less than 50%, and at 15000 m is at 20%.

In Fig.II.11.(b), by increasing the number of EDs to 300 nodes, the result PDR is less than the PDR in graph II.11.(a), seen at the increase in the number of collisions, but greater than 50% for GM and CP model, and still, RWP continues to be the worst model when it marks a PDR between 10% and 20%.



(a) (b)

Fig. II.11 – *PDR Vs Radius*

In Fig.II.12, the graph shows the effect of increasing the radius of the area on the end-to-end delay for 300 ED and one gateway; notice that the uncertainty remains constant for the three mobility models in a radius between 100 and 1000 meters, with a delay value of less than 0.06 seconds. As soon as the distance increases from 1000 meters to 15000 meters, the GM and CP models increase linearly until 0.13 seconds. The RWP model remains almost constant, keeping

the same value between 0.05 and 0.06 seconds throughout the simulation, despite increasing the radius to 15000 meters.

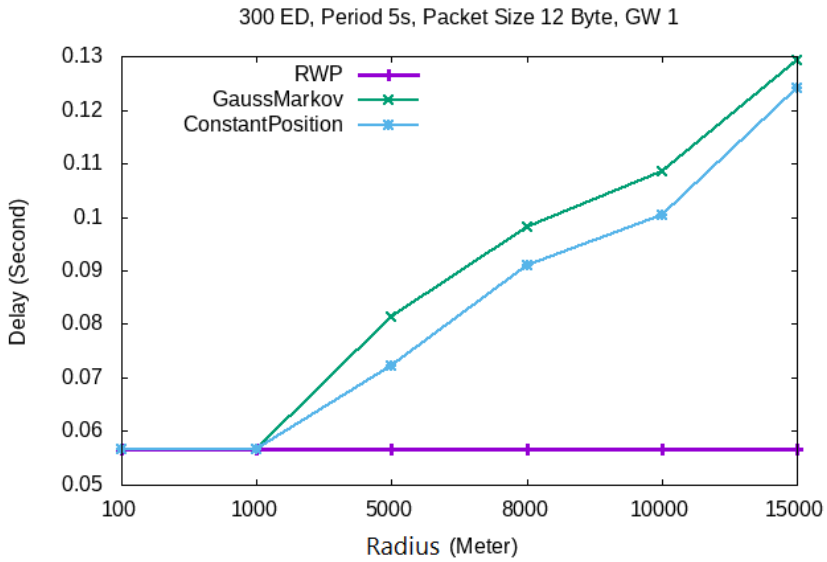


Fig. II.12 – *Delay VS Radius.*

The Fig. II.13 below shows the delay as a function of the number of nodes. Within a radius of 8000 meters, the delay increases with the number of nodes in the network, the nodes send data frames, in this case, collisions and interferences appear more frequently.

The delay increases steadily for the GM and CP model up to 160 seconds for 500 ed. As well as the delay increases slightly for the RWP which marks less than 20 seconds.

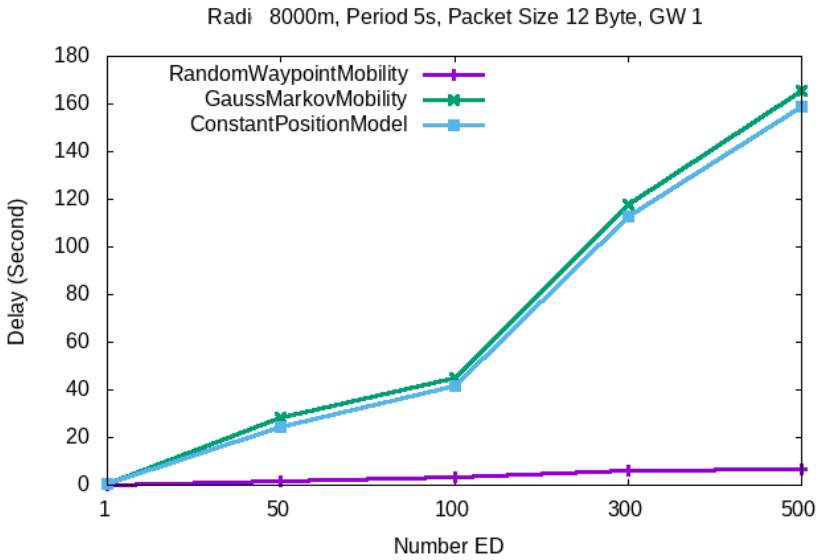


Fig. II.13 – *Delay VS Number ED.*

The graph II.14 shows the effect of changing the packet size from 12 bytes to 216 bytes on the *PDR*. We can see that the PDR decreases in a constant way with the increase of the packet size for the three mobility models. With 50 EDs and one (01) Gateway, a transmission interval of 05 seconds in a range of 20 meters we can reach a *PDR* between 75 to 85% when we have a small payload (12 bytes), keeping the same parameters, but increasing the packet size to 216 bytes, the *PDR* decreases to 30%.

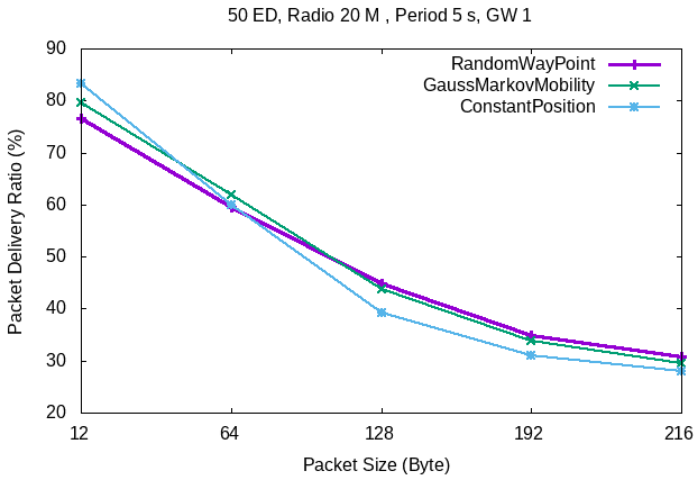


Fig. II.14 – *PDR Vs Packet Size.*

➤ **Alpha Value:**

In all the above graphs, the GM mobility model uses Alpha = 1, as shown in Table II.2, the following graph show the results when Alpha = 0.5, to better notice the impact of the random variable.

The PDR is represented in the Fig.II.15 below as a function of the number of nodes up to 500 ED, around 20 meters to the Gateway. Comparing with graph II.10 (a) (same simulation parameter used), except that the GM mobility model has changed the value of Alpha = 0.5 instead of 1, which clearly shows the impact of Alpha on the result obtained.

With Alpha = 0.5, the GM model performs better than with Alpha =1. In the graph II.10 (a), the PDR is less than 20% for the three mobility models, while with Alpha = 0.5 relative to the GM model, the PDR is greater than 50% as illustrated in graph II.15.

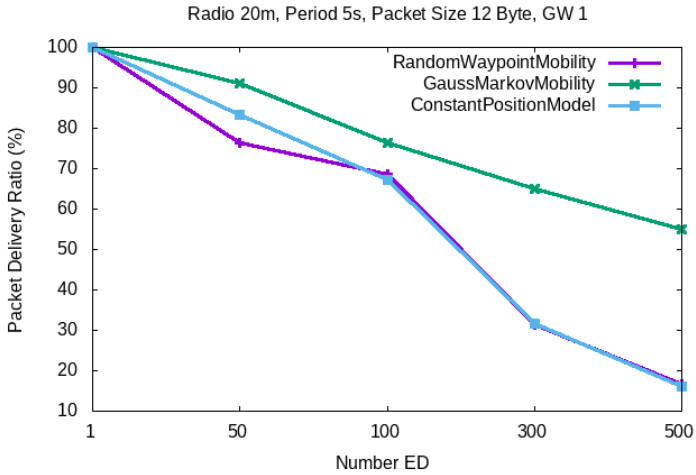
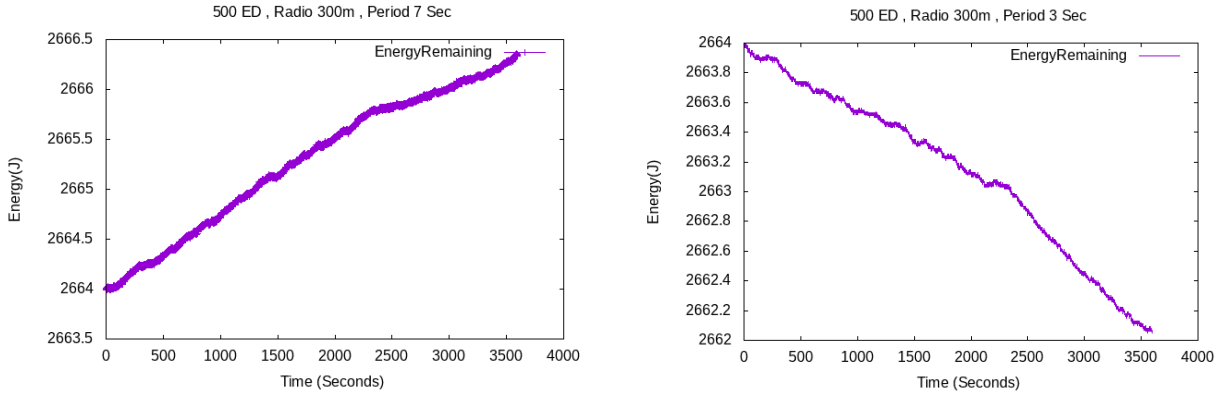


Fig. II.15 – *PDR Vs Number ED With Alpha = 0.5.*

- **Energy Harvester**

Graph (a) in picture II.16 shows the evolution of the remaining energy in the end node during the simulation time where we can see how energy recovery mitigates the energy expenditure associated with sending packets when the sending period is longer than 5 seconds (exactly 7 seconds) using the recovery module.

But when the sending period is less than 5 seconds (3 seconds), graph (b) shows the impact of the transmission speed on the energy recovery module and that the remaining energy is not replenished at all times.



(a) *Period send packet > 5s* (b) *Period send packet < 5s*

Fig. II.16 – *Impact of Period Send on Energy Remaining.*

II.8 Exepriments and Results

In this section, we describe our experimental setup, evaluation metrics and present the results of our experience.

The experiment uses the Cube Cell HTCC-AB01 module, which is based on ASR605x chips, operates at 433, 868, 915 MHz and has a maximum transmission power of 22 dBm. These modules are provided by Heltec Automation [71].

II.8.1 Cube Cell HTCC-AB01

The CubeCell products are based on the *ASR*650*x* series SiP (system-in-package) which combine a Cypress PSoC 4000 ARM Cortex-M0+ 32 bits 48 MHz MCU (with 16kB SRAM and 128 kB flash) together with a Semtech *SX*1262 LoRa tranceiver in a single package (Fig.II.17).

The CubeCell modules have an integrated LoRaWAN stack (based on Semtech's LoRaMac-node). It can be programmed via API and also via serial AT commands. The CubeCell products support development with the Arduino framework. Sketches are uploaded via the serial port. The development boards have a USB port with USB-to-serial so sketches can be easily uploaded via USB. Pinout diagrams and schematic diagrams can be found in the CubeCell documentation [71].

Fig. II.17 – *CubeCell Development board AHTCC-AB01 for arduino IOT lora node .*

II.8.1.1 Installation and start-up

After installation the *CP210x USB* to *UART* Bridge *VCP* Drivers according to windows version on the following link https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers , Let's open the Arduino *IDE* using Arduino board manager and entering the following *JSON URL* in the handler *URLs* https://github.com/HelTecAutomation/CubeCell-arduino/releases/download/V1.5.0/package_CubeCell_index.json ; as shown in Fig.II.18 :

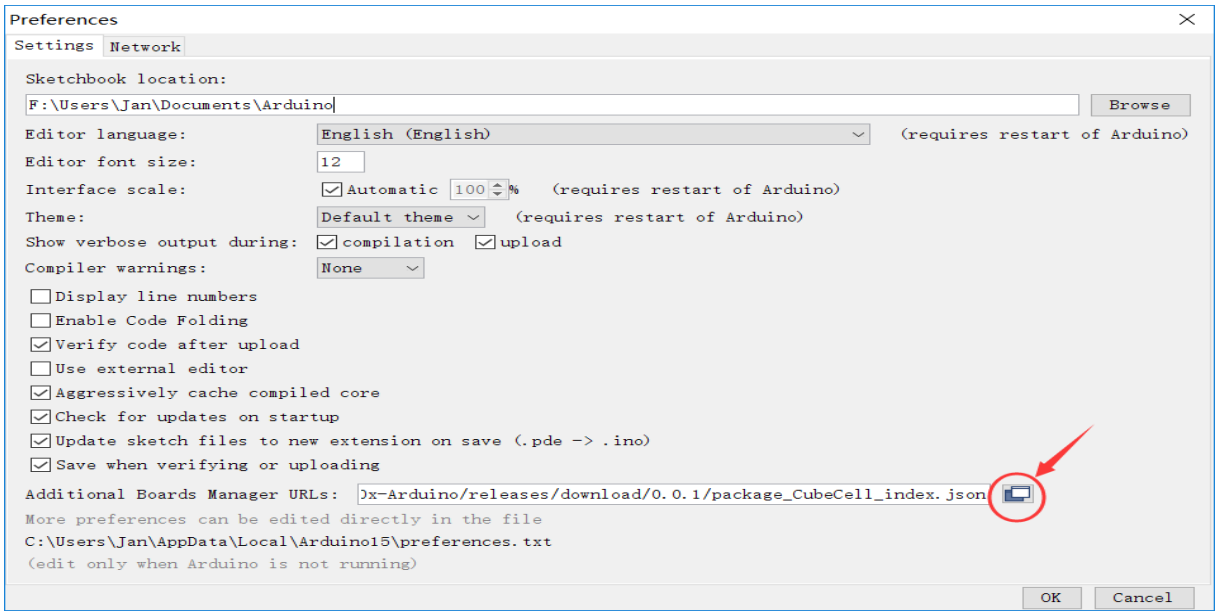


Fig. II.18 – *Package CubeCell json on Arduino Board Manager.*

For correct configuration, the LoRaWAN protocol region must be defined in the tools menu. Note that the 433 Mhz frequency is used for the Algeria region as indicated in the Fig.II.19.

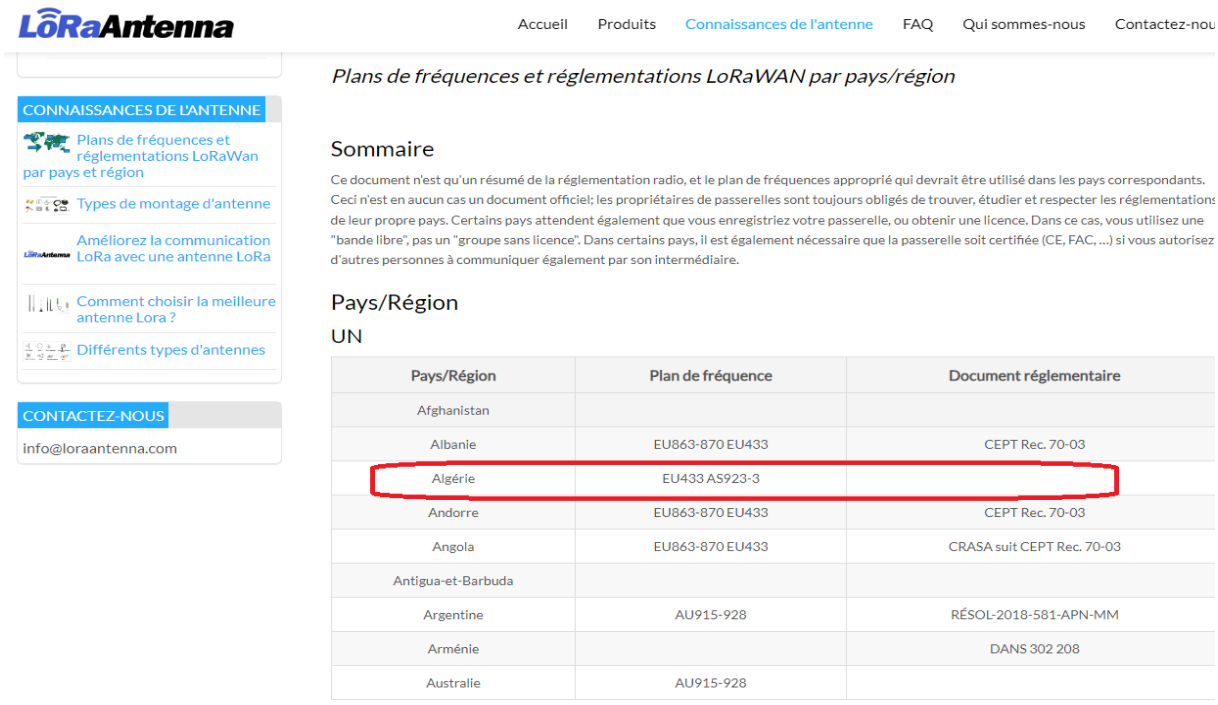
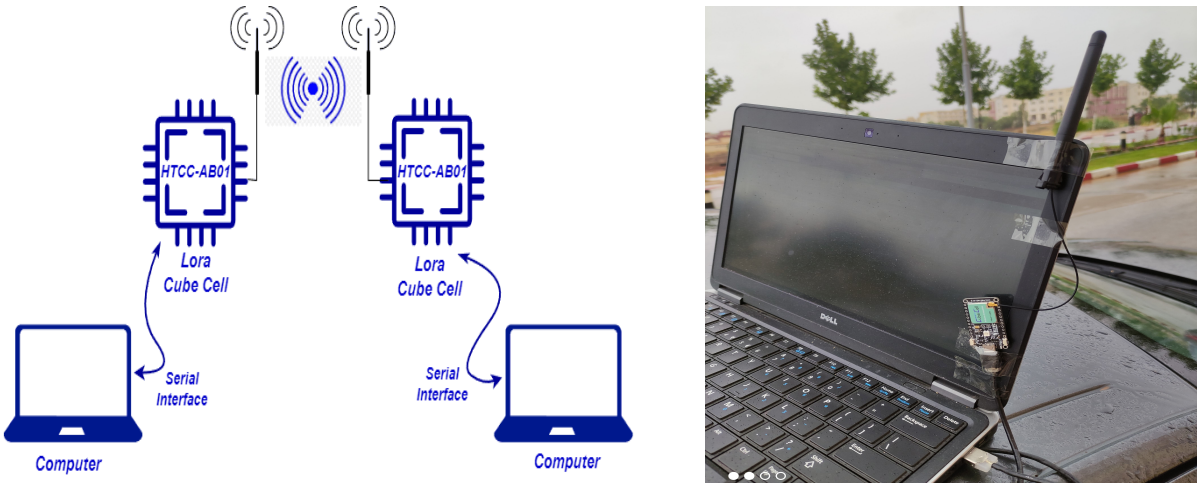


Fig. II.19 – *LoRaWan frequency plans and regulations by country [72].*

II.8.1.2 Measurement System

The measuring system, which consists of two LoRa *Cube Cell HTCC-AB01* modules and two computers, is depicted in Fig.II.20(a). As seen in Fig.II.20(b), a LoRa module is linked to a computer via a serial interface that can enable half-duplex communication and is powered by the laptop. Data packets are sent or received by the system's computers while it is operational, and are then sent between two *Cube Cell HTCC-AB01* modules through a radio link. We are able to evaluate the transmission efficiency using the measured packets.



(a) (b)

Fig. II.20 – *Measurement System.*

II.8.2 Experience results

During the experiment, the LoRa transmitter continuously sends packets of 12 bytes every 05 seconds, using two (02) $CubeCellHTCC - Ab01$ powered by Laptop one is considered as Sender fixed on the car's roof with a height of 150 Cm from the ground level as shown on Fig.II.20(b). The second module as receiver, it is mobile because it is placed inside the vehicle, the car moves at a speed between 20-70 m/s. The parameters used are illustrated in the Table.II.3.

LoRa node is set to point-to-point (P2P) transmission mode with a transmit power of 14 dBm and a 2.5 dBi gain antenna. Carrier frequency is set to 433 MHz and bandwidth is 125 kHz. The baud rate of the serial interface is set to the maximum value of 115200 bps.

Table II.3 – *Measurement configuration.*

Configure	Value
Transmit Power	14 dBm
Frequency	433 Mhz
Bandwidth	125 Khz
Packet Length	12 Bytes
Period Send	05 Sec
Baud	115200 bps

It was very difficult to find a test site for long range LoRa radio communications. Two (02) sites were recommended, the first in the city center of the wilaya of Chlef in Algeria, in an urban area at coordinates 36.1590687, 1.3239960, this is the most difficult in terms of transmission reliability because of high buildings, trees, congested areas and narrow streets. The experiment is conducted in a temperature of 31° at 23:45.

Experience indicates that the maximum distance obtained is 966.97 meters of line of sight as shown in the Fig.II.21.

Fig. II.21 – *Distance traveled in an urban area.*

Google Maps Fig.II.21 demonstrates that while the transmitter stays in its initial location and does not move during the measurement procedure, the receiver position continues to moves until it reaches the farthest point of 966.97 meters of line of sight.

The Fig.II.22 shows the experimental Received Signal Strength Indicator (RSSI) in relation to the distance travelled in metres.

The measurement of signal reception at the receiver, indicated by the value of the radio signal strength (RSSI), shows a value that continues to decrease with increasing distance.

This is consistent with the nature of long-distance communication transmission, i.e. the signal strength will be weaker as the distance between transmitter and receiver increases.

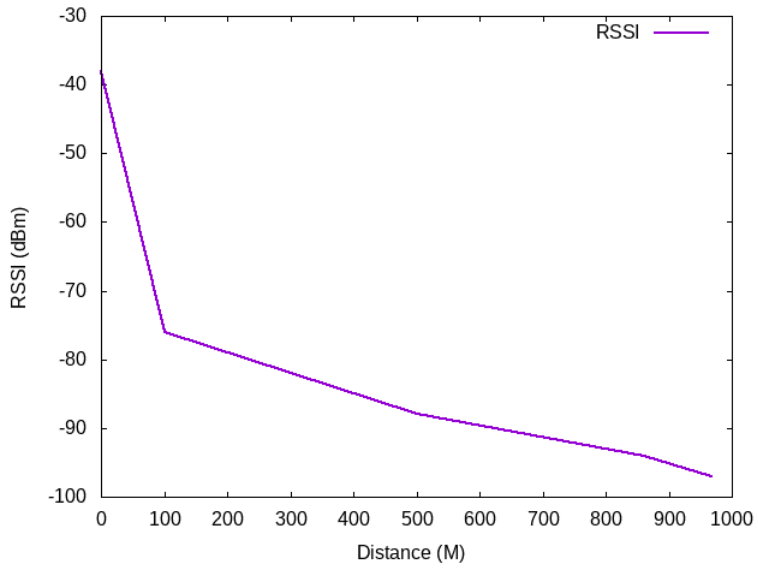


Fig. II.22 – *RSSI (dBm) vs distance (M) in an urban area.*

The farthest test was carried out at a transmitter-receiver distance of up to 966.97 m. The lowest RSSI value was -97 dBm, while the highest RSSI value was -38 dBm.

The Fig.II.23 displays the number of lost packets in relation to distance. A linear increase in incorrect and rejected packets affected by interferences, and since the lora receiver module is mobile, remoteness from the transmitter module will be followed by a decrease in signal strength.

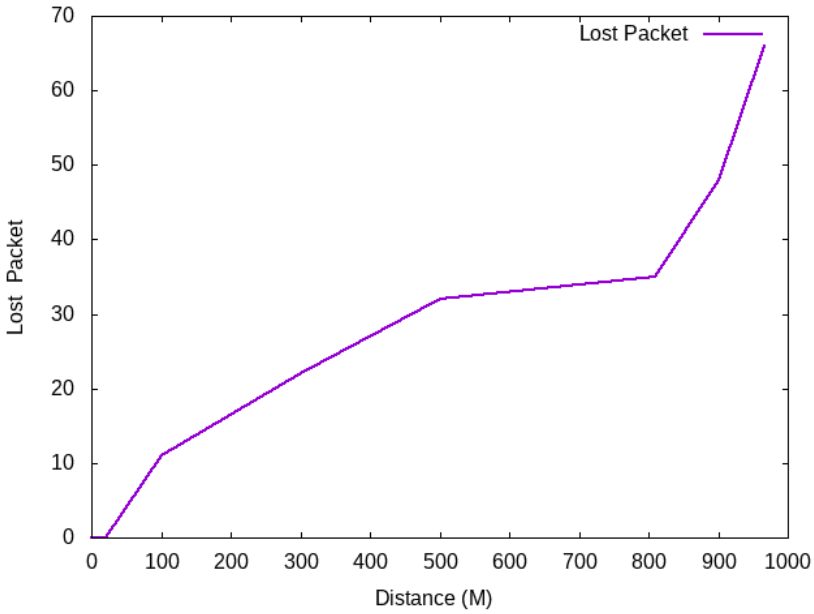


Fig. II.23 – *Lost Packet vs distance in an urban area.*

The second experiment was carried out in an open field in a rural area with fewer obstacles at coordinates 36.2486309, 1.3319770. With the same parameters, the receiver module continues to move until it reaches the furthest distance.

The location map from Google Maps and the measurement points are shown in Fig.II.24.

Fig. II.24 – *Distance traveled in a rural area.*

The experiment is conducted in a temperature of 36° at 14:30. Experience indicates that the

maximum distance obtained is 1310 meters of line of sight.

The Fig.II.25 shows the RSSI in relation to the distance travelled in metres. RSSI decreases with increasing distance. The signal strength measurement at the receiver is indicated by the lowest RSSI value of -98 dBm, the highest RSSI value of -37 dBm. We can clearly see the impact of distance on signal strength.

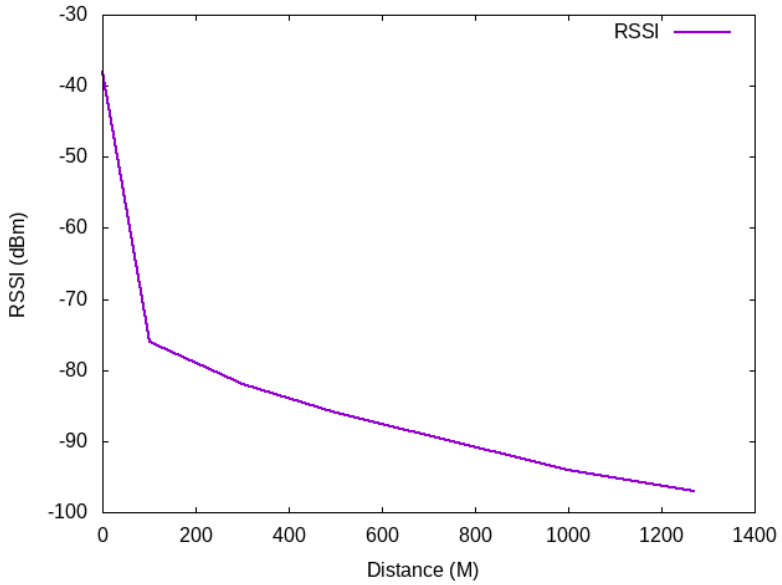


Fig. II.25 – *RSSI (dBm) vs distance in a rural area.*

The number of lost packets in relation to the distance is shown in Fig.II.26. The graph shows linear increase in lost packets. A high RSSI results in low packet loss. However, when RSSI falls below -90 dBm, packet loss becomes evident and increases rapidly as RSSI decreases.

RSSI is impacted by distance and consequently the number of lost packets is impacted by RSSI.

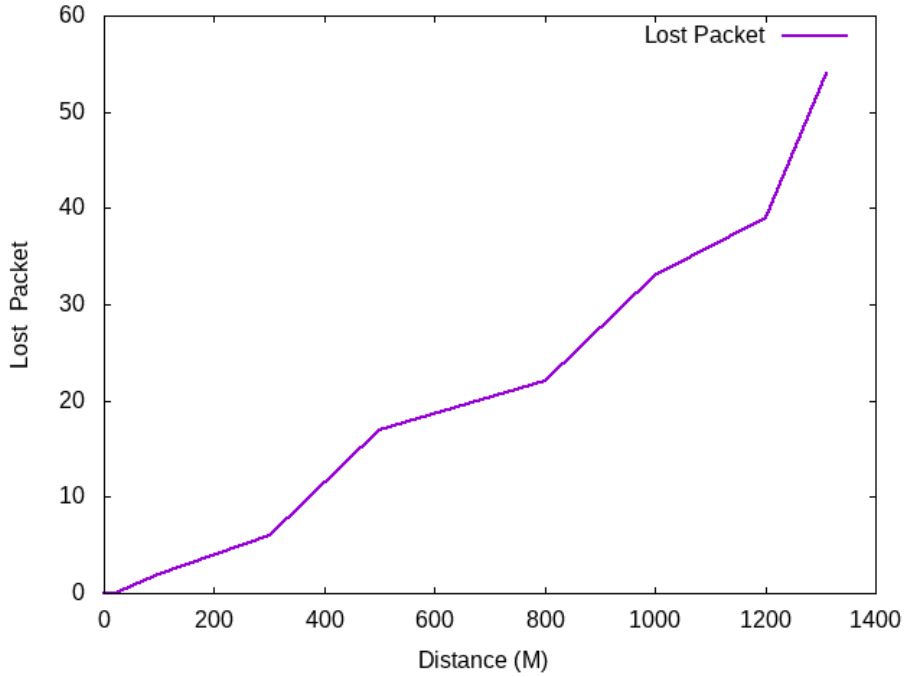


Fig. II.26 – *Lost Packet vs distance (M) in a rural area.*

II.9 Conclusion

In this chapter, we presented the first contribution of this thesis. We proposed two tools to evaluate the impact of mobility on LoRaWAN performance. We first argued our choice on the network simulator which will be used to achieve the results, subsequently for each mobility model, we simulate several scenarios, touch various performance measures such as PDR, latency, radius and packet size, increase the volume of the network by increasing the number of end devices and gateways. The simulation results indicate that the GM model performs well regarding energy consumption and radius; the GM model with Alpha = 1 performs better than the other two models, and Alpha = 0.5 performs even better in terms of PDR. At the same time, the RWP demonstrates positive results regarding delays.

Then, we take a step forward and with the same simulation parameters, we try to have real experimental results using the lora $CubeCellHTCC - AB01$ modules, measure the $RSSI$ in two different environments; rural and urban

In the following chapter, we present the spread factor allocation scheme on the LoRaWAN network in the first part, the second part of which is a state of the art on the ADR mechanism followed by a formal validation of ADR server-side through *Event-b* tools.

Spread Factor Allocation Scheme & ADR Formal Validation

III

Contents

This chapter consists of two key sections, the first section presents relation of the spread factor on Lora performance (such as PDR, Payload, Radius, network size, ToA...) using the $NS-3$ simulator and describes, examine the defies concerning the Adaptive Data Rate (ADR) mechanism proposed by LoRaWAN as well as the different variants of this mechanism presented in the literature; while the second section focuses on formal validation of the ADR algorithm.

In this thesis, we use *Event-B* and the rodin platform to formalize the ADR mechanism on the server side as the basis of our research.

III.1 Introduction

In order to enable numerous long-range packet receptions, spreading factors (SF) are crucial. Each packet is given a unique spreading factor. Thus, a modification of SF is required to enhance the data rate for transmission in areas with superior links and enable LoRa networks to adjust the trade-off between range and speed.

The number of gateways deployed and the parameters utilized for radio resources, including bandwidth, coding, spreading factors, transmission power, and others, affect the latency, resilience, and coverage of the LoRaWAN network. In the physical layer of the Chirp Spread Spectrum (CSS), a based ALOHA approach offers a variety of SFs to choose from in order to trade data rate for a long-range [73]. Thus, a higher SF sacrifices a lower data rate in order to provide a longer range, and vice versa. Different data speeds can be received by the transceivers in different channels.

The Adaptive Data Rate (ADR) scheme is a crucial LoRaWAN feature. By modifying the data rate based on the link budget for each end node in a LoRaWAN, this method seeks to reduce energy consumption and increase throughput. Bandwidth (BW), Spreading Factor (SF), Transmission Power (TP), and Coding Rate (CR) are the transmission parameters that are managed by ADR. It is split into two sections: one that operates on the end-devices (ED) and the other on the Network Server (NS). Since the majority of Class A EDs have energy limits, their ADR mechanism has been kept straightforward to prevent the need for extra computing resources, which raises energy consumption.

In the following, we first present the spread factor allocation scheme on the LoRaWAN network followed by the results obtained by the simulation using $NS-3$, then we provide a presentation of the two ADR algorithms on the End Node and Server side in order to move on to formal validation. We focus on the formal validation of the server-side ADR mechanism, using $Event-b$.

III.2 Spread Factor allocation

The term "spread factor" in the context of LoRaWAN (Long Range Wide Area Network) refers to the spreading factor used in the LoRa modulation scheme [20]. It uses chirp spread spectrum modulation, and the spreading factor is a crucial parameter in this modulation. The spreading factor determines how much the data signal is spread in time and frequency. A higher spreading factor results in a longer transmission time and better resistance to interference, but it also reduces the data rate. Conversely, a lower spreading factor allows for higher data rates but reduces the range and sensitivity to interference.

As explained previously, LoRa technology is based on spread spectrum methods, making it possible to associate a message with a chirp whose frequency varies. To enable the optimization of these transmissions, several ways of changing the signal have been implemented: these are the spreading factors. These spreading factors have been predefined and are named by a number: $SF \in \{7,8,9,10,11,12\}$. Thus, a message sent at SF12 will go further than that sent at SF7, as shown in Fig.III.1. However, this involves several disadvantages. First, messages with a higher spreading factor will be longer. Indeed, with spreading spectrum, spread the message as if it were more prominent, making the transmission time longer, the time in the air longer, and therefore a higher risk of collision. In addition, generating a longer message means more energy must be consumed to transmit the message.

To understand the impact of the spreading factor, the simulation can note that if a node emits at SF7 for one year every ten minutes, its battery can last one year, while a node talking at SF12 will not exceed 0.8 years or 292 days [19]. Choosing the appropriate spreading factor depends on the specific requirements of IoT applications. It needs to consider factors such as the distance between devices, the desired data rate, and the environment's level of interference.

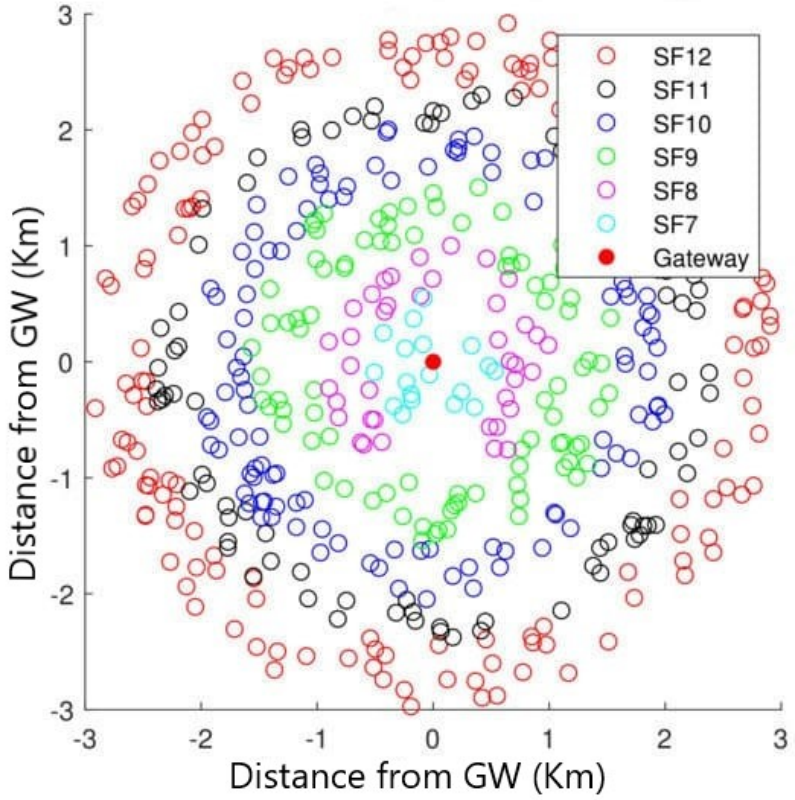


Fig. III.1 – *Distribution of nodes according to spread factor.*

The bit rate decreases clearly when the SF increases. This will imply that the larger the SF, the longer it will take to send a packet. And so, the sensor will consume more energy to send a packet. Therefore, this parameter has a lot of importance in energy consumption; a too small SF will prevent the packets from arriving at their destination. While too large, an SF will significantly increase energy consumption (because the transmission time increases greatly with SF). This increase in transmission time also increases the risk of collision. In addition, the choice of SF is essential because other sensors can emit simultaneously, generating a collision. The sensor will, therefore, have to deliver the lost packet. Choosing an SF that is not too used by other sensors will help avoid collisions and, therefore save energy.

III.2.1 Similar Works

In the literature, the ADR scheme has been modified and implemented to satisfy different LoRaWAN objectives by targeting network performance metrics. In this section, we present several lines of research and studies relevant to LoRaWAN performance and QoS. Some research has focused on evaluating the performance of the LoRaWAN protocol in terms of scalability, battery lifetime, coverage and Time on Air. Other work has aimed to improve the performance of the LoRaWAN protocol, by improving the ADR or enhancing the speed of transmissions UL.

In [74] a simulated environment is implemented to assess how well SF assignment schemes work. Additionally, a brand-new intelligent SF assignment strategy that optimizes SF assignment using machine learning methods such as Decision Tree Classifier (DTC) and Support Vector Machine (SVM) is proposed. In terms of Packet Delivery Ratio (PDR), it is noted and demonstrated that the suggested smart SF assignment approaches produce encouraging simulated results.

This study [75] presents an improved energy consumption model for a LoRaWAN node based on in-situ data. The number of network nodes, the collision probability, which is depen-

dent on the sensor density, and the number of retransmissions are all taken into account by this revised model. Results reveal that the number of sensors that may be incorporated into a LoRaWAN network is constrained owing to the probability of collision and show the influence of the number of nodes in a LoRaWAN network on the energy consumption of a node.

Asad Ullah et al.[76] offer a technique based on K-means clustering that addresses the issue of LoRa spreading factor (SF) allocation. They evaluate the network performance while taking the unconfirmed-mode class-A LoRa Wide Area Network (LoRaWAN) model's outage probabilities into account. The suggested technique makes flexible use of SF allocation by allowing for various user distributions over SFs. Such distribution results in application-specific network parameters. Results from simulations show how the number of nodes in each SF and the distance from the gateway affect transmission reliability by taking into account various network situations and realistic factors. When compared to the baseline model, theoretical and simulation studies demonstrate that the SF allocation strategy increases the network's average coverage probability by up to 5 percentage points.

Authors [77] propose two SF allocation techniques to improve the packet success ratio by reducing the effect of interference. The first method, known as the channel-adaptive SF recovery algorithm, increases or decreases the SF based on the ED packets' re-transmission to reflect the network's channel condition. In the second method, SF is distributed to EDs according to their sensitivity at first deployment. Using thorough simulations that take into account channel interference in both LoRaWAN confirmed and unconfirmed modes, these methods are validated. They demonstrate through simulation results that the SFs have been applied in an adaptive manner to each ED, and the suggested methods improve the packet success delivery ratio in comparison to the conventional SF allocation schemes.

This study [78] presents a new SF assignment structure intending to improve performance overall. The assigned SFs in the proposed method, in particular, are no longer determined by the distance but rather by the instantaneous channel realizations. Compared to traditional SF assignment techniques, such a dynamic assignment of the SFs among LoRa users dramatically improves overall performance. Through numerical simulations, the suggested system is assessed in terms of Symbol Error Rate (SER).

A distributed genetic algorithm-based method, or meta-heuristic, is developed in [79] to address the problem at hand. However, by allocating the spreading factor to EDs while keeping in mind the average energy consumption of all EDs, the technique presented can improve LoRaWAN performance. In contrast, the techniques shown in the existing studies provide the average energy consumption as a result of the performance improvement by the spreading factor allocation. Numerical examples demonstrate a technique's utility. Although the technique's average energy consumption across all EDs is lower than that of the approaches suggested in the existing studies, the technique's packet reception probability (PRP) performance is superior to that of the strategies demonstrated in the existing studies.

they [80] examine the effects of node and gateway scaling and densification, accounting for the capture effect, on system reliability. They put forth an optimization issue to determine the node distribution at various spreading factors (SF) in LoRaWAN networks with several gateways. After that, they present an adaptive technique that makes it simple to apply SF optimization by modifying the signal-to-noise ratio thresholds. The performance of the suggested algorithm is also contrasted with that of legacy LoRaWAN and other state-of-the-art methods. Throughput and packet delivery ratio of the network are improved by the suggested method, according to simulation findings, which greatly beat state-of-the-art algorithms.

Authors [81] provide Energy Efficiency of LoRaWAN EE-LoRa , a methodology to boost the energy effectiveness of LoRaWAN networks with multiple gateways by spreading factor selection and power control. First, they optimize the network's energy efficiency, which is determined by comparing the throughput to the amount of energy used. The key to solving this issue is figuring out which spreading factor's node distribution is the best. To reduce the transmission power at nodes without compromising the dependability of communications, power

control is then used in the second stage. Compared to historical LoRaWAN and related state-of-the-art algorithms, the simulation results demonstrate that the proposed algorithm significantly increases the energy efficiency of LoRaWAN networks.

The Authors [82] aim to provide a framework for investigating the viability of extensive downlink communication in LoRaWAN with moving End Devices (EDs). They test two straightforward ways to minimize packet loss when EDs switch from one inside range gateway to another, starting from a real-world implementation of a Firmware Update Over the Air application. Then, using simulation, they enhance the ns-3 LoRaWAN module to expand the study. These tests demonstrate that packet loss and the amount of time required to finish transmitting the data to the EDs are significantly influenced by the approach adopted and the placement of the gateways.

Sorensen et al [83]. have introduced an SF allocation approach called equal load-based SF allocation to improve the network performance. In this SF allocation scheme, the end devices are allocated SFs such that the packet traffic load per SF is the same. Based on the different SF allocation schemes, the authors have estimated single LoRaWAN cell dimensions. They have shown the analytical and MatLab simulation comparison for three SF allocation schemes (uniform, distance-based, and equal load-based). For equal load SF allocation scheme simulation, the authors considered a single LoRaWAN cell with uneven end device distribution. According to the simulation, the coverage probability and network throughput for equal load-based SF allocation are higher than the distance-based SF allocation.

With the same objective as [84], Amichi et al. [85] have addressed an approach for optimal allocation of Spreading Factors with a target to enhance the network throughput under the consideration of co-SF and inter-SF interference. The researchers have maximized the minimum attainable average bit rate in LoRa Networks by proposing an SF allocation algorithm based on a many-to-one matching algorithm. The proposed work has the scope to extend the work by jointly controlling transmission power and SFs allocation scheme through a distributed algorithm. The simulation results indicate that the developed SF allocation approach performs better than the Random and Distance-based SF allocation schemes in terms of data rate and network throughput.

In order to achieve data rate fairness and minimize collisions between the end nodes, [86] proposes the Fair Adaptive Data Rate algorithm (FADR), which computes data rate and TP assignment. In order to optimize DER across all end nodes, the method makes use of RSSI values for determining SF and TP allocations. The authors suggest using a "region concept" to allocate SFs based on the RSSI, with SFs being assigned within the regions in accordance with the given proportions. By allocating low TP to end nodes with a weak signal and high TP to end nodes with a weak signal, the proposed power allocation strategy optimizes the RSSI of the end nodes and ensures a fair distribution of resources.

By using low TP levels, they reduce power consumption by 22% compared to traditional ADR and maintain the end node's lifespan while achieving uniform DER for all end nodes regardless of their distance from the gateway. The authors demonstrated that the fairness index between end nodes falls as the number of end nodes in the network increases. As a result, end nodes closer to the gateway would be more likely to communicate than those farther away. As a result, this strategy works best in very small networks with end nodes placed close to the gateway. It is worthwhile to explore the use of various propagation models in the simulation.

The authors proposed a retransmission-assisted ADR (RM-ADR) in [87] to improve the PDR by reducing the retransmission attempts of the ED in UL. Their results show improved performance in PDR, energy consumption, and convergence period compared to the existing state-of-the-art ADRs mechanisms.

III.2.2 Simulation results

In this section, the study presents the results of different simulation scenarios by varying the number of end devices, payload, and radius. These results are analyzed and evaluated in order to study the relation between different spread factors on PDR, ToA, and radius in Lora Networks.

The study considers a situation up to 10, 100, 200,300, and 500 ED with 01 GW, The simulation time is equal to 3600s (01 hour). The node sends data frames to the GW (uplink) with a size of 12 bytes every 5 seconds. The ED positions are randomly assigned around the GW in a radius of 1000, 5000, 8000, 10000, 15000, 20000 and 25000 meters. evaluation and comparison of efficiency carried out using the $NS-3$ simulator. The simulation of the network with an increasing number of nodes with the different values of the SF as indicated in the Table III.1.

Table III.1 – *Simulation Parameters.*

Parameter	Value	Unit
N of Nodes	10, 100, 200 , 300 ,500	-
Radius	1000,5000,8000,10000,15000,20000,25000	Meter
Period	5	Second
Packet Size	12	Byte
GateWay (GW)	1	-
Simulation Time	3600	Second
Spread Factor	SF7 , SF8 , SF9 , SF10, SF11 , SF12	-
BandWidth (BW)	125	Khz
Code Rate (CR)	4/5	
Simulator	NS3 (Version 3.35)	-
Operating System	Ubuntu 24.4 64 bit	-

Fig.III.2 shows the impact of the distance from the gateway on PDR, with the distribution of 300 ED around GW, the graph shows that when the radius is equal to 1000 m the PDR of SF7 is higher than the rest of the SFs than 30% followed by SF8 of 15% and decreases as the distance from the GW increases. While the rest of the SFs are null; is this is due to the fact that they are out of range, however SF9 increased from 5000m and SF10, SF11, and SF12 begin to increase from 8000 m which clearly explains the distribution of nodes illustrated in the Fig.III.1.

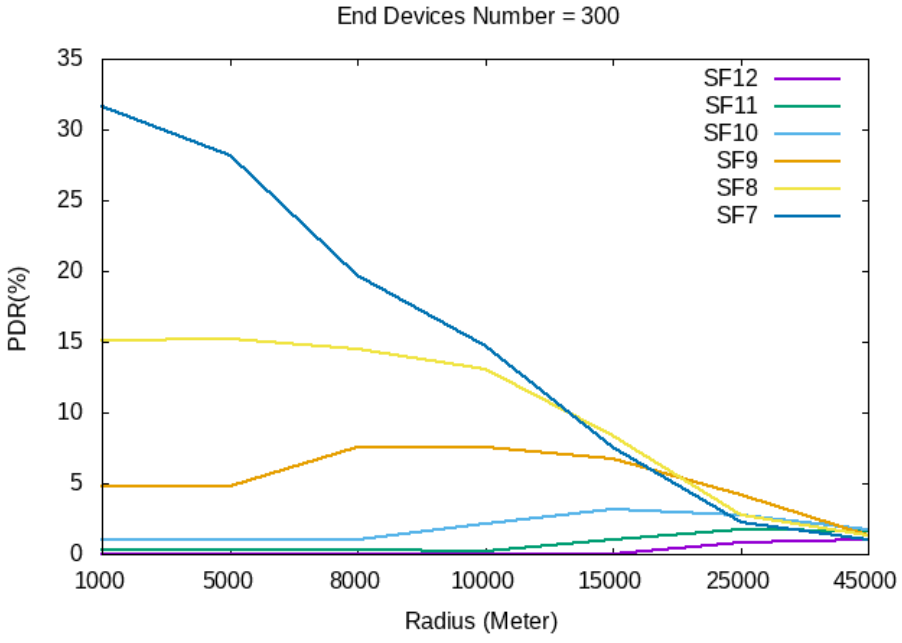


Fig. III.2 – *PDR Vs Radius .*

In Fig.III.3, the graph shows the effect of increasing the Number of ED on the PDR for radius = 8000 meters and one Gateway. Increasing SFs increases air time. This increases the number of collisions thus decreases the PDR.

High SF schemes gives poor PDR results as the number of nodes increases. SF10 has the highest PDR of over 80%, and SF7 and SF12 have the lowest PDR of less than 50% due to the distance and to the remoteness of the gateway, which represents the shortest and longest distance respectively, which further argues the graph in Fig.III.2

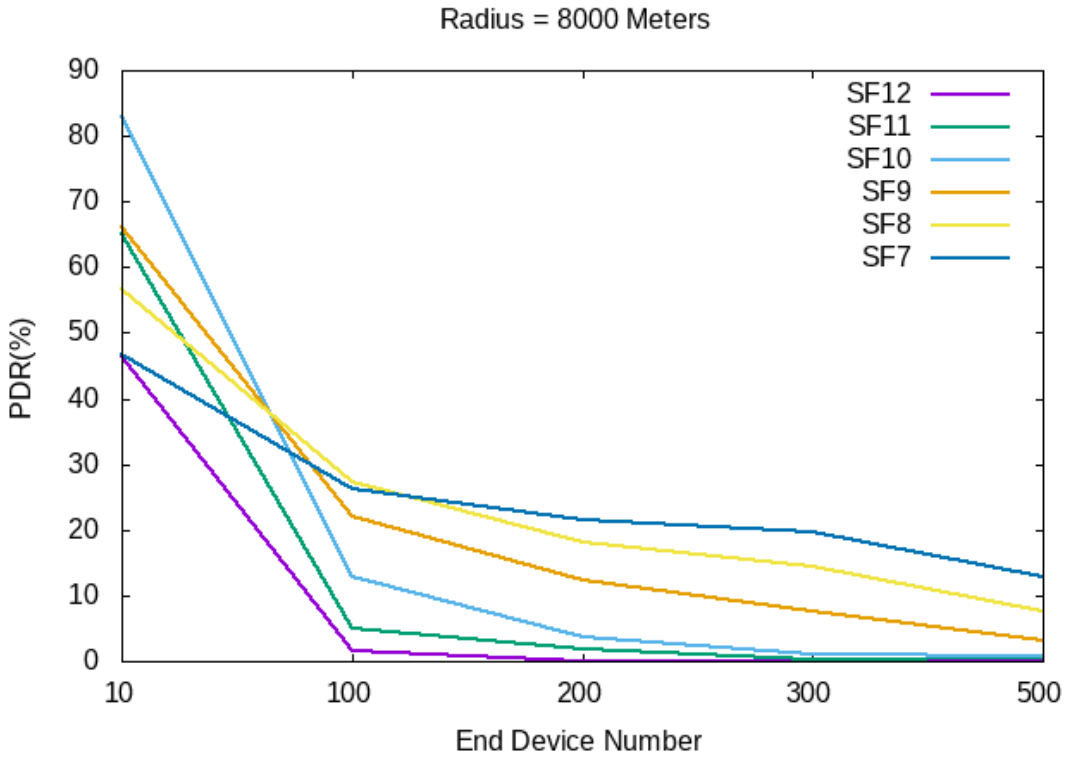


Fig. III.3 – *PDR VS Number ED.*

In LoRa, each symbol is presented as a sinusoidal signal, the frequency of which changes within the bandwidth(BW) around the central frequency (fc). The chirp rate depends only on bandwidth. The duration of these chirps $T_{symbol}(SF)$ depends on both spreading factor and bandwidth as computed in equation I.2. The chirp duration directly depends on the spreading factor on which the device is transmitting.

A higher SF means longer communication distance with a lower data rate, resulting in longer air time of transmitted packet as shown in Fig. III.4. Date rate R_b can be calculated as shown in equation I.1. Time on air for different configurations for each packet can be calculated using a formula I.6 provided in LoRaWAN specifications [3].

Fig III.4 demonstrates the significant impact of time-on-air, payload, and spreading factors on the packet with varying payload from 5 to 50 bytes transmitted from nodes to the gateway. The findings indicate that the ToA increases with goods, which was resulting in a slower transmission speed. Additionally, it has been noted that the ToA increases with increasing SF; spreading factor 12 has the longest time-on-air because of the distance between nodes and gateways. A slight variation in time-on-air was seen between the nodes that employed spreading factors of 7 and 8, indicating that the distance between those two spreading factors and the gateway was about similar for each channel that was sent.

It concludes that a high SF can cause great latency; the longer the time-on-air results in less

data transmitted per unit of time with the same bandwidth. However, a high SF will be used in some cases where an increase in coverage is desired.

Longer packet transmission time makes LoRa more vulnerable to threats where multiple transmissions are happening at same time using same SF and frequency channel or in cases where there is high power transmission going on in parallel to low power transmission which might dominate the low power signal.

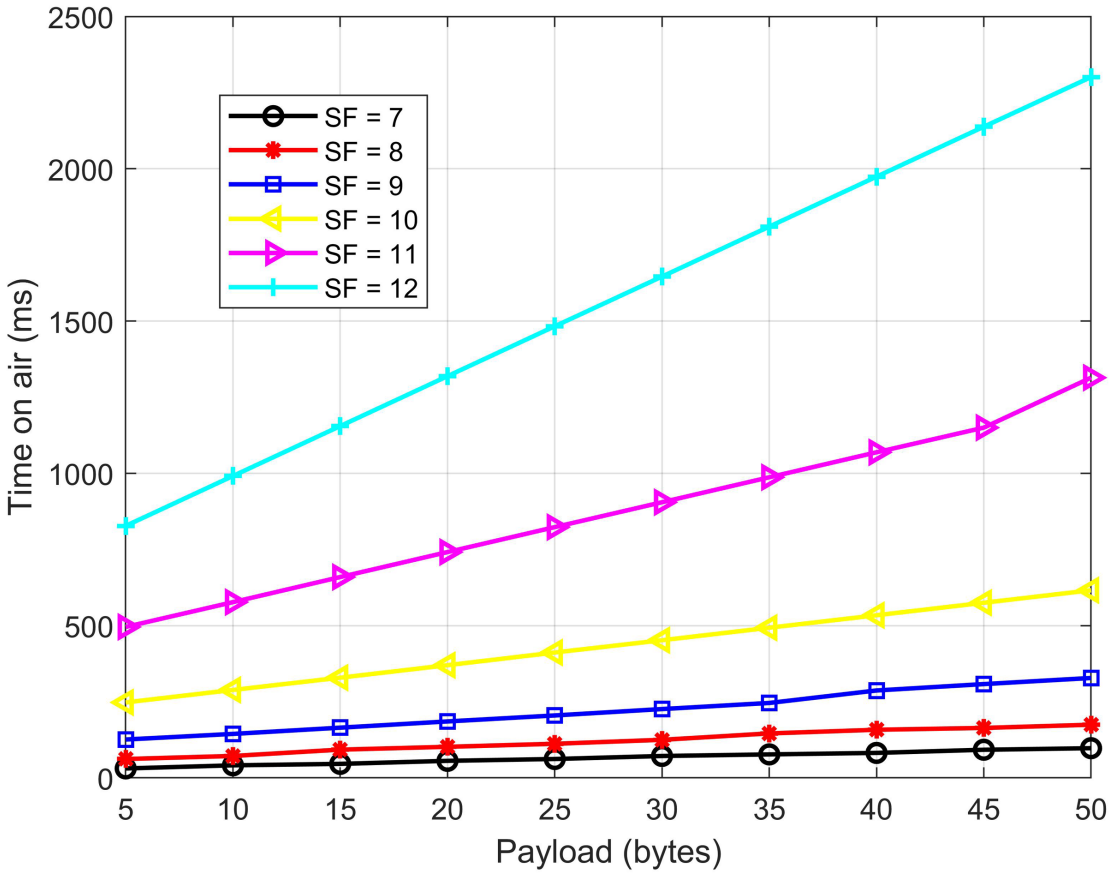


Fig. III.4 – *Time on air Vs Payload with BW = 125 Khz, CR = 4/5.*

III.3 Adaptive Data Rate (ADR)

In order to achieve an optimal operating mode in terms of network performance (reliability, range, delay, throughput and energy consumption), LoRaWAN offers the possibility of using different throughput by manipulating the parameters mentioned previously (SF, BW, CR and TP) according to the needs of the application. [88] defined 6720 combinations of these parameters giving rise to different performances. To meet the performance needs of the network on the one hand, and to minimize energy consumption (and therefore maximize the lifespan of the nodes' batteries) on the other. Furthermore, the use of ADR significantly increases the capacity of such a network, since the data packets that are transmitted using different SFs are orthogonal and can be transmitted concurrently [40].

To increase the packet delivery ratio (PDR), an ADR scheme was created for LoRaWAN that may control the transmission settings of the end nodes. From the end node to the gateway, the ADR regulates the transmission parameter settings for the UL data. Based on the link budget estimation in the UL message and the maximum SNR needed for precise packet decoding at the current data rate, the ADR algorithm is in charge of controlling the data rate and transmission power of end nodes.

"Network-managed ADR or Static ADR" refers to the situation where the NS controls the ADR based on the history of UL packets received by fixed end nodes. Mobile end nodes cannot use the network-based ADR method due to channel attenuation that happens when the device moves. When it comes to mobile end nodes, ADR is carried out "blindly" on the end node side, a process known as "Blind ADR." Four (04) distinct commands for the ADR are present in the LoRaWAN MAC layer, as seen in Table III.2.

Table III.2 – *LoRaWAN Adaptive Data Rate (ADR) Commands.*

Command	Description
ADR	End node sets this bit requesting the gateway to control its data rate.
ADR = 1	NS will control the end nodes data rate
ADR = 0	NS will not control the end nodes data rate
ADRACKReq	Allows EDs to periodically receive confirmation that NS is receiving UL msgs
ADRACKReq = 1	NS should respond to confirm receipt of UL data
ADRACKReq = 0	Confirmation of receipt of UL data not required.
LinkADRReq	Transmitted by NS to request end node to change its transmit parameters
LinkADRAns	Transmitted by nodes in response to LinkADRReq command
LinkADRAns = 1	Transmit parameters successfully set
LinkADRAns = 0	Command is discarded

Initially, during the process of joining a LoRaWAN network, the node often uses SF12 (the largest radio range) to ensure it reaches a gateway (or a random SF value, depending on the node manufacturer). After receiving 20 successive packets, applies ADR-Server (Algorithm 1) to choose an appropriate SF (smaller than the initial SF used by the node) based on SNRs (Signal to Noise) values in an attempt to optimise data throughput, ToA and consequently reduce power consumption. In the following we present the two ADR algorithms (ADR-Node and ADR-Server).

III.3.1 ADR Server side

ADR scheme provides optimum data rate, ToA, and energy consumption [22]. The data rate is highly dependent on SF and BW. An end device must set the ADR flag in the up-link packet format to enable the ADR functionality. Once the ADR is set, the Network Server (NS) can control and instruct the end device for transmission parameters. The ADR scheme dynamically adapts the transmission parameters based on the previous performance history of each node to enhance the system throughput and capacity (last 20 packets received). For instance, consider the same network topology as shown in Fig.I.8.

In Algorithm 1, The network server collects m most recent up-link packets transmission data (like Data Rate and SNR) for an end device A. The NS takes the maximum received SNR and corresponding data rate value out of m received up-link packets SNR values. The maximum received SNR value is called $SNR_{measured}$. The NS calculates the margin as:

$$Margin = SNR_{measured} - SNR_{limit} - Margin_{default}$$

.

where, SNR_{limit} and $Margin_{default}$ are defined for each SF value. Based on the calculated Margin value, the NS suggests the optimum data rate and TP to the end device for the next transmission [89].

After calculating the SNRmargin, a number of steps $Nstep$ is calculated to determine the number of times we will decrease the SF or increase the TP., ADR should be iterated to adjust both SF and TP as:

$$N_{step} = round(Margin)/3$$

- If N_{step} is negative and maximum transmission power is not reached, the network server increases the transmission power by $3 \times N_{step}$, until the maximum transmitted power is achieved (TXmax= 14 dBm in Europe).
- If N_{step} is positive and the minimum SF is not yet reached, the server decreases SF (increases DR). When the lowest SF has already been attained, the server considers decreasing TP until its value reaches the minimum value (TXmin=2 dBm).

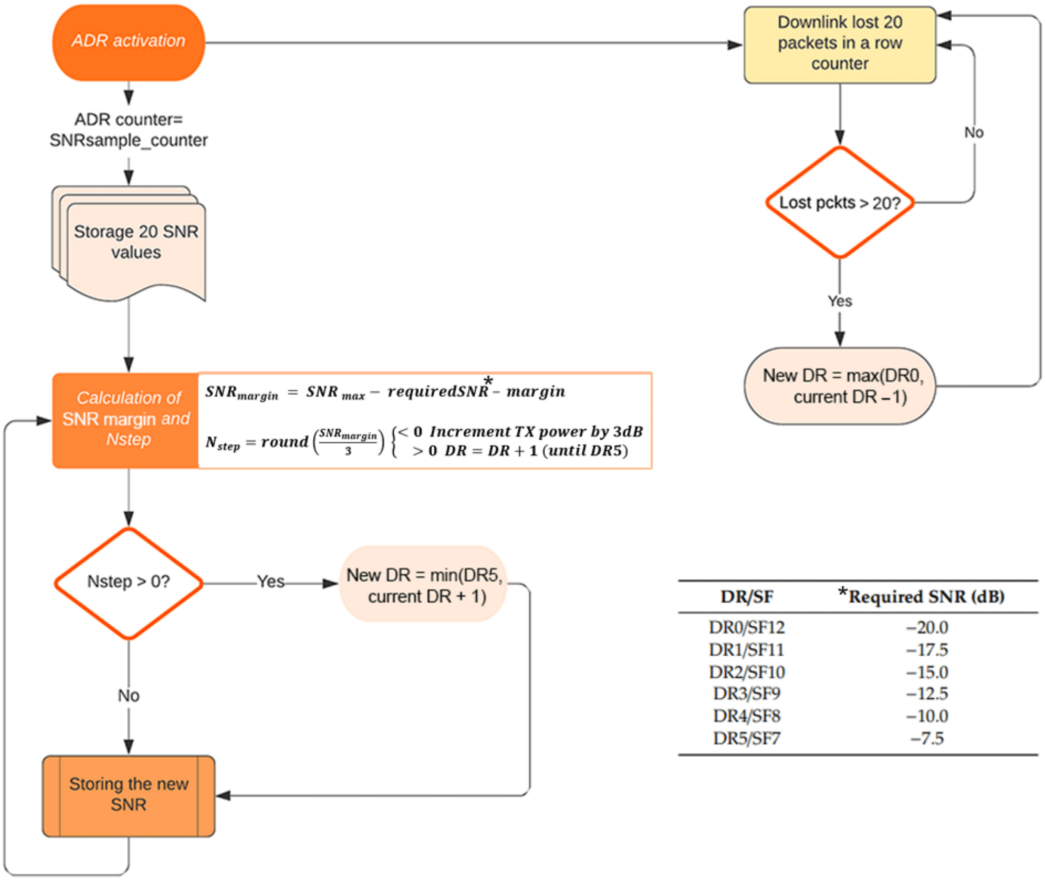
Finally, SF and TP parameters are sent to the ED via a DL MAC command *LinkAdrReq*. This mechanism on the server side is illustrated in the diagram III.5.

Algorithm 1 ADR Server side

```
for every received packet do
    SNRmargin ← (SNRmax − SNR(DR) − margindb)
    NStep ← int(SNRmargin/3)
    if NStep = 0 then
        Exit
    else
        if NStep > 0 then
            if DR < DR5 then
                DR ← DR + 1
            else
                TxPower ← TxPower − (3db)
            end if
            Nstep ← Nstep − 1
            if TxPower = min then
                Exit
            else
                Go to Begin
            end if
        else
            if TxPower < max then
                TxPower ← TxPower + (3db)
                NStep ← Nstep + 1
                Go to Begin
            else
                Exit
            end if
        end if
    end if
end for
```

III.3.2 ADR End Device side

An end node configures the ADR bit in the frame header to inform the gateway that it needs to use ADR. After ADR is set up, the NS utilizes the MAC command *LinkADRReq* to regulate the data rate and TP of the end node. To indicate acceptance or rejection of the updated settings, the end node will reply with the *LinkADRAns* command. In order to allow



DR/SF	*Required SNR (dB)
DR0/SF12	−20.0
DR1/SF11	−17.5
DR2/SF10	−15.0
DR3/SF9	−12.5
DR4/SF8	−10.0
DR5/SF7	−7.5

Fig. III.5 – *Adaptive Data Rate (ADR) flow implemented in the Network Server [89].*

end nodes to periodically confirm that the NS received the UL message, the ADR algorithm includes an acknowledgement system. In an effort to reconnect, the end node will switch to a reduced data rate if it does not receive an ACK message. Using the ADR system that is nested on the end node side, the end nodes can additionally control the ADR transmission settings. This implies that both the NS side and the end node side of the ADR scheme can operate concurrently.

According to the latest release of the LoRaWAN standard (v1.1)[3], the ADR algorithm 2 at the end node is simple, is only to increase the range by increasing the SF if the data does not reach the gateway (lost transmissions).

Algorithm 2 ADR End Device side

```
1: if Transmission UL then
2:     ADRACKCNT ← ADRACKCNT + 1
3:     if ADRACKCNT == ADRACKLIMIT then
4:         SF ← SF + 1
5:         if Transmission DL received then
6:             ADRACKCNT ← 0
7:         end if
8:     end if
9: end if
```

There are two parameters that have been specified, namely ADR_ACK_LIMIT and ADR_ACK_DELAY. These parameters have been set to 64 and 32 by default, respectively. For every UL transmission, a *ADRACKCNT* counter that defends the number of retransmissions

of an unacknowledged packet is triggered. The ADR_ACK_CNT counter is raised by one for each UL packet sent by an end node. The end node sets the *ADRACKReq* bit and waits for an ACK from the gateway for the next ADR_ACK_DELAY UL packets after the ADR_ACK_CNT equals ADR_ACK_LIMIT (= 64 for EU 868) without any DL response.

The end node must reduce the data rate by raising the SF for wider range in an effort to restore network connectivity if there isn't an ACK before the ADR_ACK_DELAY UL message. According to the most recent release, end nodes first raise TP in order to ensure secure connectivity. The end nodes then lower the data rate as part of the next step if that is insufficient [88]. Upon receiving an acknowledgement, this counter is reset. The flow of the ADR scheme implemented at the end node is shown in Fig.III.6.

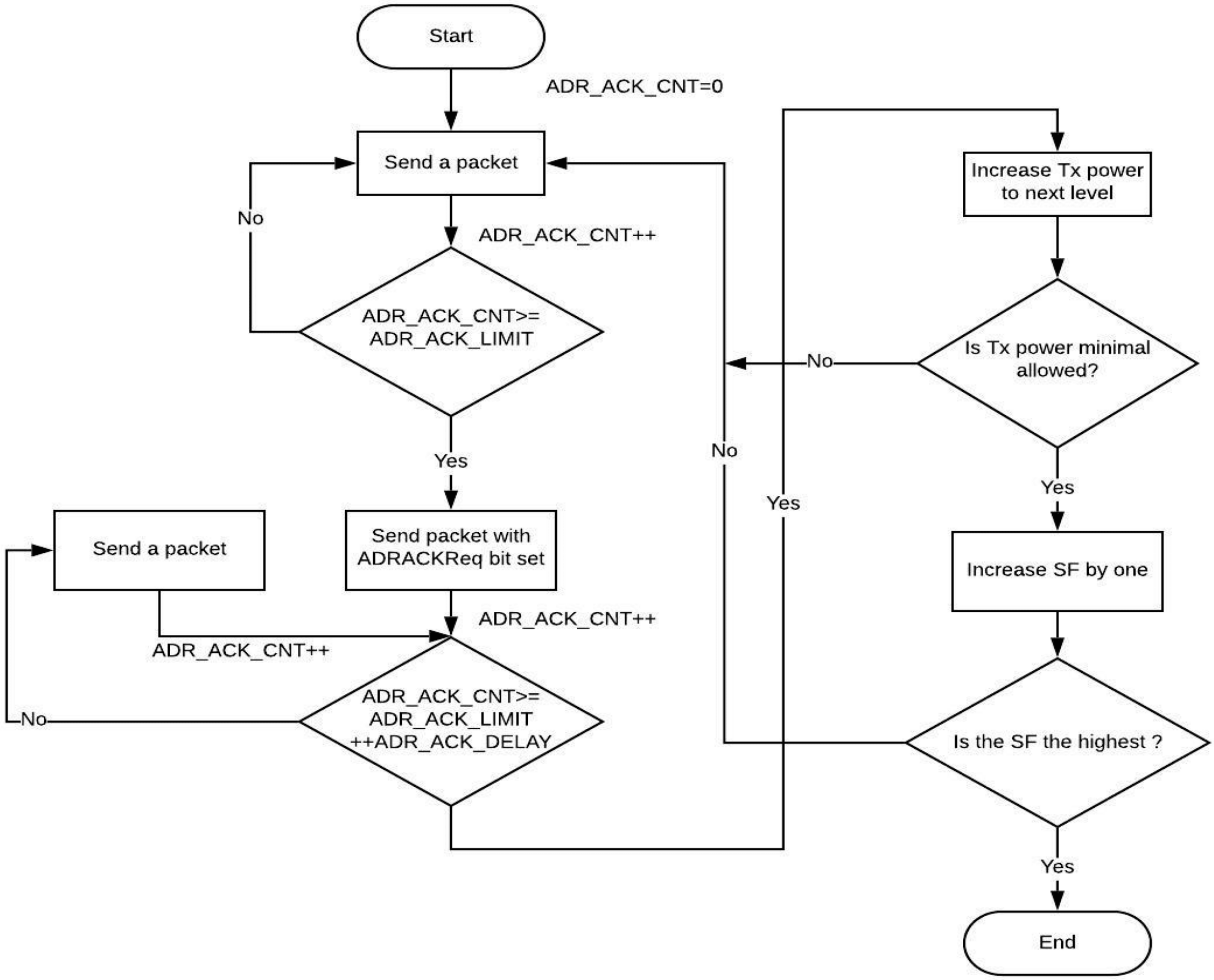


Fig. III.6 – *Adaptive Data Rate (ADR) flow implemented in the end node.*

III.3.3 Blind ADR (B-ADR)

LoRaWAN uses an energy-efficient adaptive data rate (*ADR*) to distribute resources to EDs, which is advised for static applications. In contrast, a *ED* and a *GW* significantly alter the channel scenario for mobile Internet of Things (*IoT*) applications [90]. Consequently, it is not advised to use standard *ADR* in these circumstances [91]. Semtech recently proposed a Blind ADR (B-ADR) to improve battery life and geographical coverage for mobile applications like pet monitoring. to improve battery life and obtain adequate coverage [91]. As seen in Fig.III.7 [91], three data rates will be employed in place of a single data rate, with a distinct periodicity based on the data rate.

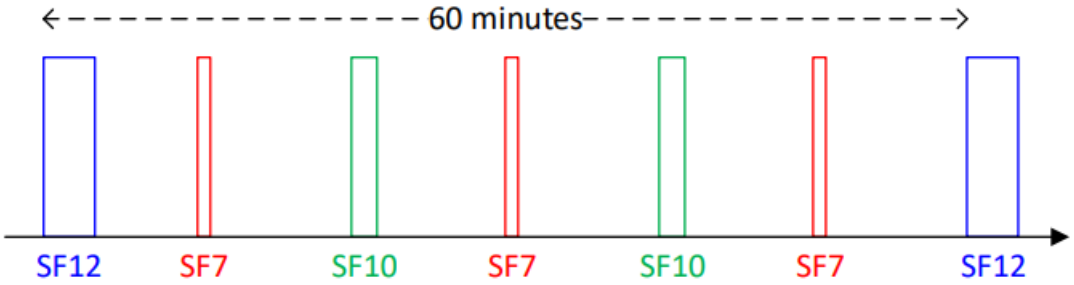


Fig. III.7 – *Blind Adaptive Data Rate (B-ADR).*

The Blind ADR (*B-ADR*) uses three SFs:

- SF 12 (once), every hour
- SF 10 (twice), an hour
- SF 7 (three times), an hour

For pet tracking application, this gives us frequent outdoor position information when the pet is moving, with fewer updates when the animal is located indoors and therefore, less mobile. An overview of B-ADR is provided and the limitations highlighted and examined in this study [92]

In addition, despite the different simulations done and the results obtained, which support specification [3], we believe it is more appropriate to use mathematical and logical methods to validate the ADR mechanism. Therefore using the formal method to verify and validate the ADR protocol in the LoraWan network. We used Event-B (the formal method) to model the layers of the protocol and their properties and Event-B invariants to ensure the consistency of the protocol, and we will add more guarantees to the validity of the protocol. For now we focus on the formal validation of the network server side ADR mechanism.

III.4 Formal validation

Unlike classical development approaches, formal methods use the mathematical logic to rigorously reason about the correctness of a construction. They provide a strong assurance of the absence of bugs in the software. However, they are generally expensive in resources and thus, they are currently reserved for the development of safety-critical systems [93].

In this scetion, we present the formal validation of the ADR algorithm, network server side using Event-B and Rodin platform, Before we get into the specifics of our formal validation of the ADR protocol, it's important to remember some fundamental concepts about the formal methods used in this thesis, especially the Event-B method.

III.4.1 Formal methods

Formal methods are a modelling technique used to adequately define or describe a set of requirements or specification. Formal methods are a mathematically formed (hence the term formal is derived) descriptive requirement used in order to precisely determine the correct intended functionality fully [94]. Furthermore, formal methods also refer to a set of techniques and tools which are used to design and verify software and hardware systems. More often the term "mathematically rigorous" is associated with the description of formal methods; this term refers to well-formed statements in a mathematical logic.

Formal verification, similarly, adopts rigorous deductions in logic. Mathematical rigorous approach presents a set of statements based on a rule of inference and hence can be physically checked, for example, negation of a negation is equivalent to the original expression [94]. Activities like system specification, specification analysis and proof, program verification, and transformational development are included in formal methods. They utilise mathematical notation and mathematical proofs.

III.4.1.1 Event-B formal method

Event-B [95] is a formal method for specifying, modeling, and reasoning about systems based on set theory and predicate logic invented by Jean-Raymond Abrial. It can be used to model all sorts of discrete event systems, among them sequential programs [96]. Event-B evolved from Classical B and Action Systems. On the one hand, Event-B is a simplification as well as an evolution of the B-Methods; on the other hand, Event-B is influenced by the action systems approach. It has the same structure as an action system, which describes the behavior of a reactive system in terms of the guarded actions that can take place during its execution; it is supported by the Rodin platform. Compared to Classic B, Event-B has the capacities to model a system. System development with B-method proposes a cycle as illustrated in the Fig.III.8 [93].

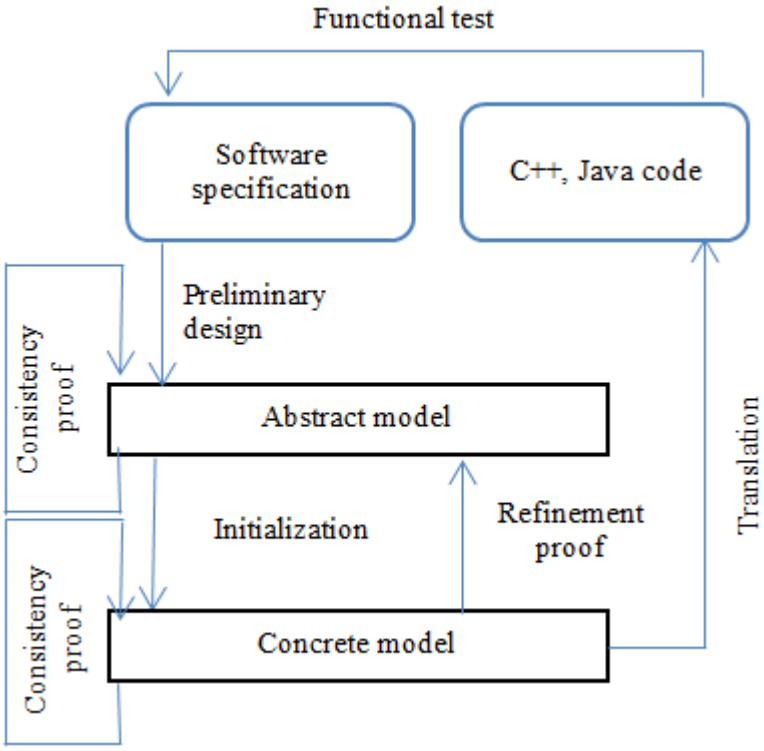


Fig. III.8 – *B-method methodology [93].*

III.4.1.2 Event-B structure and notation

Machines and contexts make up a model in Event-B [97]. The static (types and constants) and dynamic (variables and events) components of a model are found in contexts and machines, respectively. The axiomatic properties of an Event-B model are provided by contexts, while the behavioral properties are provided by machines. Modeling elements are things that belong to machines and situations. The connections between machines and surroundings vary. A context can be "seen" by machines and "expanded" by other contexts. Referring to contexts

as its static element, a machine can be "refined" by other machines [98]. As seen in Fig.III.9, the link between machine and context.

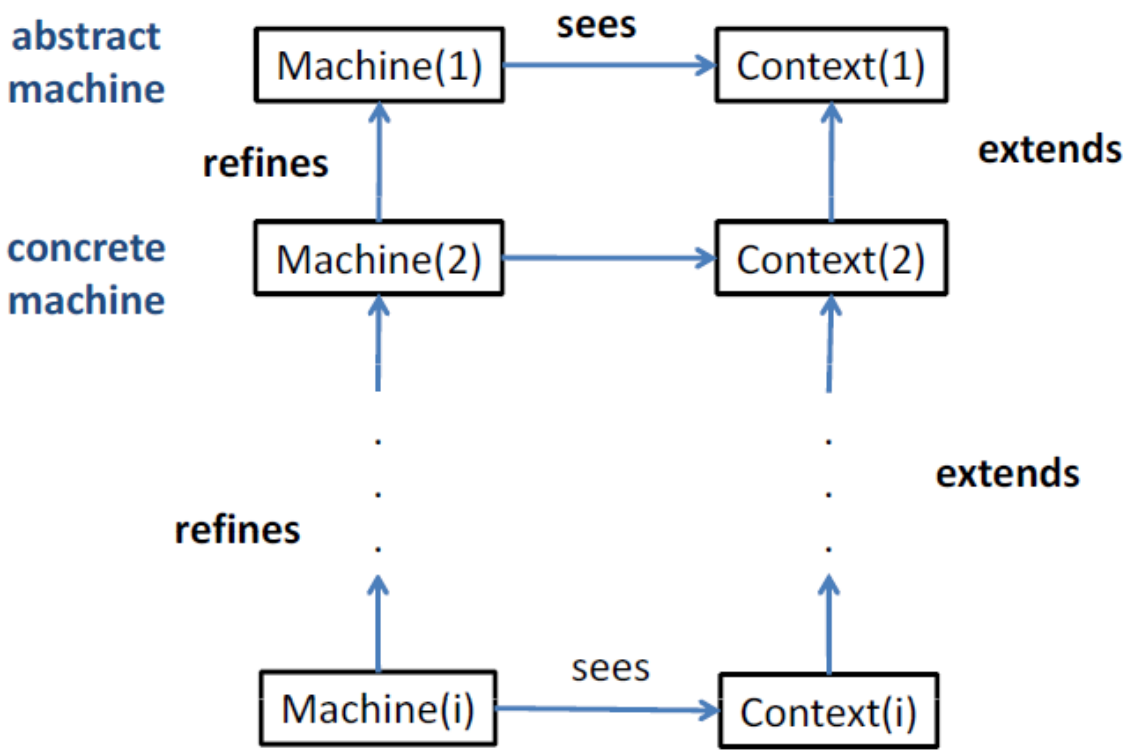


Fig. III.9 – *Machine and context relationships [94].*

From a given machine, *Machine*1 in this case, a new machine, *Machine*2, can be built as a refinement of *Machine*1. In this case, *Machine*1 is called an abstraction of *Machine*2, and *Machine*2 is said to be a concrete version of *Machine*1.

An event-B project has the following components:

1. **Context:** It is a type of first-order theory that contains declarations of constants and axioms about these constants using an abstract language based on first-order logic predicates, simple sets theory, and arithmetic on the Z set of integers. For example, the context for searching a value in an array of strictly positive integers can be described as follows:

CONTEXT ..... Array
CONSTANTS
n array size
T the array
v value to search
AXIOMS
axm1: $n \in \mathbb{N}_1$
axm2: $T \in 1..n \rightarrow \mathbb{Z}$
axm3: $v \in \mathbb{Z}$
END

2. **Abstract machine:** it describes the dynamic structure or the system evolution using constants and axioms imported from context by the SEES clause. The abstract machine must describe the following:

- The state of the system through a set of variables;
- The consistency of the state by a set of invariants (some formulas to satisfy);
- Some events define the possible evolutions of the machine's state.

For the abstract machine associated with the example of searching in an array, we can write:

MACHINE S1
SEES Array
VARIABLES
result1
Elmt
INVARIANTS
inv1: $result1 \in \mathbb{Z}$
inv2: $Elmt \in \mathbb{Z}$
EVENTS
Initialisation
begin
act1: $result1 := 1$
act2: $Elmt := 0$
end
Event evt1 $\langle$ordinary$\rangle$ $\widehat{=}$
when
grd1: $n \geq 10$
then
act1: $Elmt := 2$
end
END

3. **Events:** In Event-B, the execution of an event modifies a model's state. A name, a collection of guards $G(t, v)$, and a few actions $S(t, v)$ make up each event, where t are the event's parameters and v is the system's state, which is determined by variables. Every event is atomic and can only be carried out while its guards are in place. Only one of the events is selected nondeterministically to be carried out when the guards of many events are held simultaneously. Table III.3 lists the three possible formats for an event. In its most basic form, an event will only have a few actions; in its second form, it may consist of guards and actions without parameters; and in its third form, it will include guards, actions, and some parameters [98].

Table III.3 – *Event forms.*

Three Possible Forms of an Event
E = begin $S(v)$ end
E = when $G(v)$ then $S(v)$ end
E = any t when $G(t, v)$ then $S(t, v)$ end

One of the various types of assignment that can be applied to an event's action is shown in the table. III.4. A variable is denoted by x, an expression by $E(t, v)$, and a predicate by $P(t, v, x)$. There is determinism in the first assignment form. The assignment in the second row is not deterministic; for example, it assigns a value that is not in the empty set. The third row is regarded as non-deterministic since it gives x a value based on the criteria that was specified.

Table III.4 – *Action forms.*

Type	Generalized Substitution
Deterministic	x := $E(t, v)$
Non-deterministic	$x :\in E(t, v)$
Non-deterministic	x :| P(t, v, x́)

III.4.2 Refinement in Event-B

In an Event-B development, we are advised to build the system in a sequence of consecutive layers, beginning with an abstract representation of the system, instead of having a single huge model. The abstract model ought to offer a straightforward perspective of the system, emphasizing its primary function and salient characteristics. The abstraction ignores the specifics of how the goal is accomplished. The system's functional details are progressively added to the abstract model in a stepwise fashion. We refer to this procedure as refining.

Context extension and machine refining are two methods for improving an Event-B model. It is feasible to add new sets, constants, and properties while keeping the existing ones while taking context extension into account.

There should be a minimum of one observable event for every abstract one. Multiple concrete events have the ability to refine an abstract event. This is known as event splitting. Additionally, more than one abstract event might be refined by one concrete event. Event merging is the term for this.

III.4.3 Rodin Platform

Rodin is the name of the tool platform for Event-B. It allows formal Event-B models to be created with an editor. It generates proof obligations that can be discharged either automatically or interactively. The Rodin Platform [99] is an open tool set implemented on top of Eclipse, makes formal modelling possible using Event-B. It is devoted to supporting the development of such systems. It has been developed within the framework of the European project Rodin [95] (2004–2007).

It contains a modeling database surrounded by various plug-ins: static checker, proof obligation generator, provers, model-checkers, animators, UML transformers, requirement document handler, etc. The database itself contains the various modeling elements needed to construct discrete transition system models: essentially variables, invariants, and transitions [100, 101]. Fig.III.10 shows an overview of the opening window of the Rodin GUI.

III.5 ADR Formal validation

In this section, we have used the Rodin platform (*version 3.7.0*) to model and prove the correctness of the *ADR* protocol by detailing the corresponding event-B model for each specification.

We first make precise the overall purpose of the *ADR* protocol:

Specifications:

- *SPEC01: Each network device's transmission power and data rate must be within their valid ranges based on the link quality.*
- *SPEC02: The transmission power and data rate of each device in the network must be adjusted based on the quality of the link to the gateway.*
- *SPEC03: Each device in the network can transmit at most one message per channel per transmission window.*

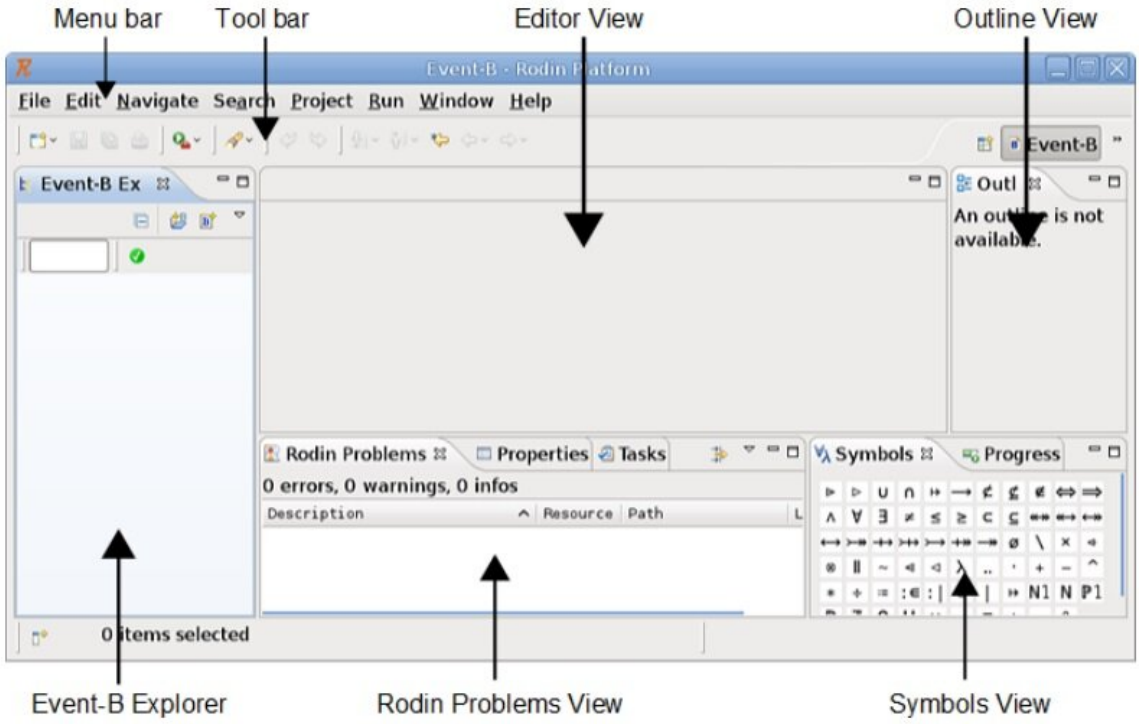


Fig. III.10 – *Overview of Rodin GUI.*

- *SPEC04: Each device in the network must wait for a random period before transmitting a message.*
- *SPEC05: The number of devices transmitting on the same channel during the same transmission window must be limited to avoid collisions.*
- *SPEC06: The gateway must be able to receive messages from the devices by using the same data rate and transmission power as the device.*
- *SPEC07: All devices are either active or inactive.*
- *SPEC08: All devices and gateway have IDLE or Sending or Receiving states.*

All of these written specifications and requirements will be changed into algebraic and mathematical notation, as we will see later: Such a model contains two parts, context and machine. The static part of the *ADR* protocol is defined by the following context, named context_datarate in event-B :

CONTEXT .....datarate
SETS
Devices
Channels
Windows
TransmissionPower
DataRate
StateDevice
Mode
Gateways
Link_Status
END

➢ Where the StateDevice and Mode represent the status of devices and gateways, respectively.

CONTEXT .....datarate
CONSTANTS
maxDataRate
minBatteryLevel
maxTransmissionPower
minTransmissionPower
n
Sending
Receving
IDLE
Link_Up
Link_Down
END

➢ LinkQuality is the quality of the link between a device and the gateway.

CONTEXT .....datarate
AXIOMS
axm1: $maxDataRate = 8$
axm2: $minBatteryLevel = 1$
axm3: $maxTransmissionPower = 7$
axm4: $minTransmissionPower = 0$
axm6: $n \in \mathbb{N}$
END

This context describes the ADR protocol , a device can be in one of sending, receiving, and IDLE modes, shown by the AXIOMS closure in the previous context. also, we model the status of links by two states linkup and linkdown as indicated in the following axioms:

axm7: $partition(Mode, \{Sending\}, \{Receving\}, \{IDLE\})$
axm9: $partition(Link\_Status, \{Link\_Up\}, \{Link\_Down\})$

The dynamic part of the ADR protocol is defined by an Event-B machine named DataRate, for which we must give sets of variables, invariants, and events.

The first part defines devices parameters and variables (transmission power, data rate,BatteryLevel, and link quality), different modes or statuses of devices, and gateways: receiving, sending, and IDLE. Finally, the variable $RSSI$ represents the measure of signal power level. The rest of the model—the INVARIANT and INITIALIZATION closures describe respectively possible and initial values of variables.
The machine DataRate is illustrated below :

MACHINE .....DataRate
SEES datarate

VARIABLES

tranmissionPower
dataRate
BatteryLevel
linkQuality
Active
Inactive
r
Link
mode
RSSI
Link_Dev_Gtw
SF
RxDelay
modeGatway

INVARIANTS

inv3: $tranmissionPower \in \{0..7\}$
inv4: $dataRate \in \{0,1,2,3,4,5,6,7\}$
inv5: $BatteryLevel \in \{0,1,2,3,4,5\}$
inv13: $Active \in BOOL$
inv14: $Inactive \in BOOL$
inv15: $r \in 0\,..\,n$
inv6: $linkQuality \in (Devices \rightarrow Link\_Status)$
inv7: $BatteryLevel \geq minBatteryLevel$
inv8: $(minTransmissionPower \leq tranmissionPower)$
inv9: $(maxTransmissionPower \geq minTransmissionPower)$
inv10: $((dataRate \leq 1) \Rightarrow (tranmissionPower \leq minTransmissionPower))$
inv11: $dataRate \leq maxDataRate$
inv12: $(\forall d,c,w \cdot (d \in Devices \wedge c \in Channels \wedge w \in Windows)) \Rightarrow (\forall w \cdot w \in Windows)$
inv16: $Link \in Devices \leftrightarrow Devices$
inv17: $mode \in (Devices \rightarrow Mode)$
inv18: $RSSI \in \{0,1,2,3,4,5,6,7,8,9\}$
inv19: $((RSSI = 0) \Rightarrow (tranmissionPower = minTransmissionPower))$
inv21: $(\forall d,c,g \cdot (d \in Devices \wedge c \in Channels \wedge g \in Gateways)) \Rightarrow (\exists w \cdot w \in Windows)$
inv22: $\forall d \cdot (\exists g \cdot g \in Gateways \wedge d \in Devices) \Rightarrow (\exists w \cdot w \in Windows)$
inv23: $Link\_Dev\_Gtw \in Devices \leftrightarrow Gateways$
inv24: $((RSSI > 1) \Rightarrow (tranmissionPower > minTransmissionPower))$
inv25: $\forall a \cdot (a \in Devices \wedge mode(a) = IDLE) \Rightarrow linkQuality(a) = Link\_Up$
inv28: $modeGatway \in (Gateways \rightarrow Mode)$
inv26: $SF \in \{7,8,9,10,11,12\}$
inv27: $RxDelay \in \{1,2\}$

END

After many tests, all these variables are set to the empty set, and the corresponding value as an initialization event. In event-B using Rodin, the signalization machine for the transport layer is coded as follows:

Initialization
act3: $tranmissionPower := 0$
act4: $dataRate := 0$
act5: $BatteryLevel := 1$
act6: $linkQuality := Devices \times \{Link\_Up\}$
act7: $Active := TRUE$

act8: $Inactive := FALSE$
act9: $r := 0$
act10: $Link := \varnothing$
act11: $mode := Devices \times \{IDLE\}$
act12: $RSSI := 1$
act13: $Link\_Dev\_Gtw := \varnothing$
act14: $SF := 8$
act15: $RxDelay := 1$
act16: $modeGatway := Gateways \times \{IDLE\}$

A set of events controls how nodes work together or change how they act. For each event, some conditions, called guards, need to be checked, and then a list of possible actions for the node to take is made based on the situation.
For example, in the event $UpdateLinkQuality$, $grd1$ and $grd2$ are used to ensure that both the device and the gateway are in "$IDLE$" mode so that the link between them can be updated. A node must permit its links to be updated if guard conditions are verified.

In Rodin platform version 3.7.0, all properties and events are added, and after a lot of tweaking and fixing, we have fixed all syntax and meaning errors, as shown in Fig.III.11.

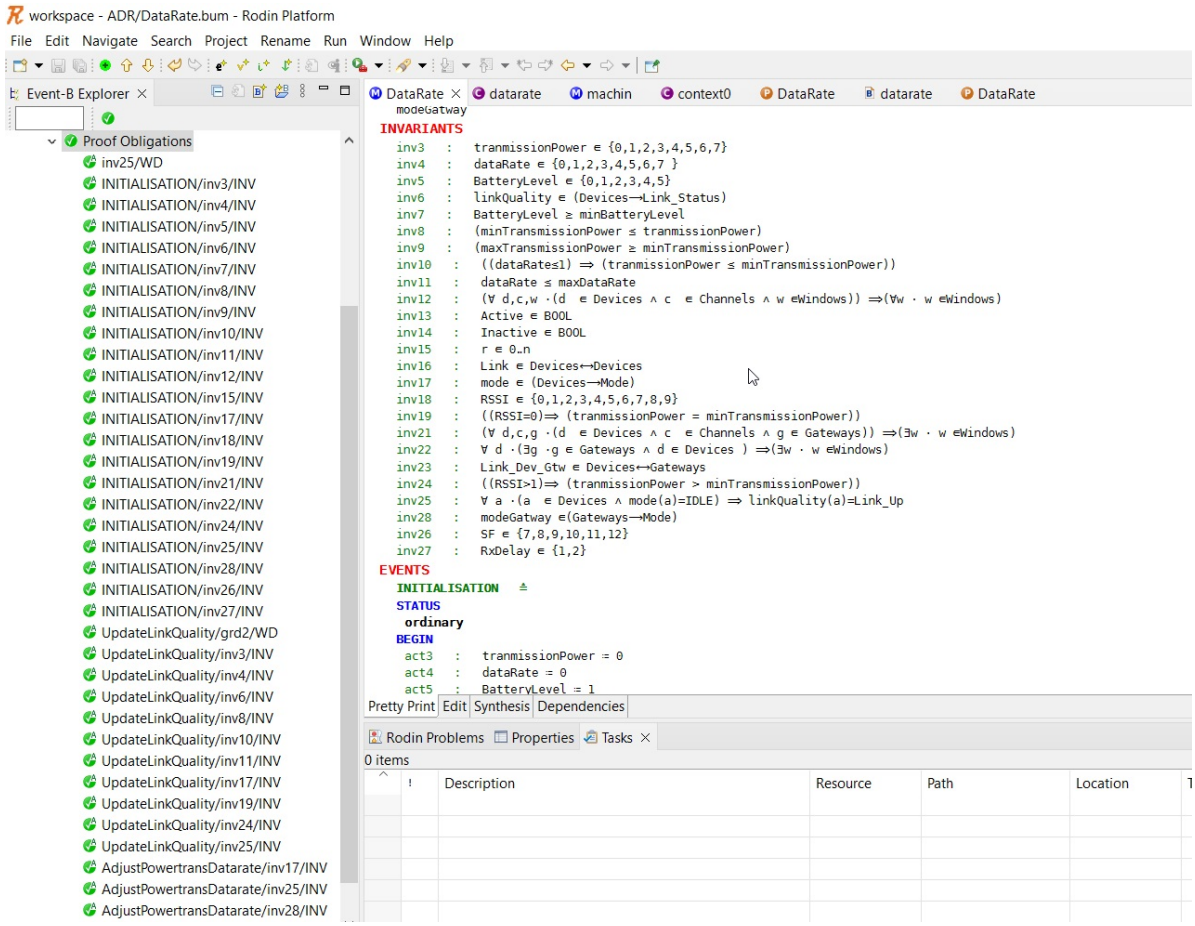


Fig. III.11 – *Rodin Platform.*

Event-B allows to describe the behavior of a system by a chain of refinements of abstract machines. This automatic refinement is based on simple rules ensuring the obtaining of a refinement: introduction of a new convergent event, refinement of an event by several (one to many), introduction of a new variable, reinforcement of guard, reinforcement of invariant and instantiation of a local parameter of an event by a state variable.

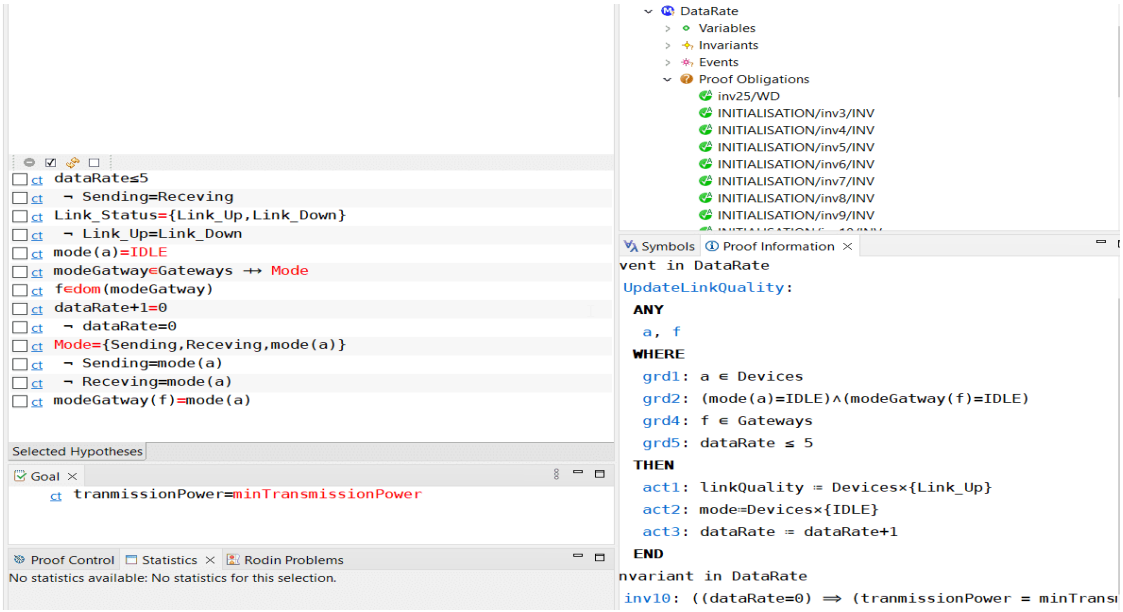
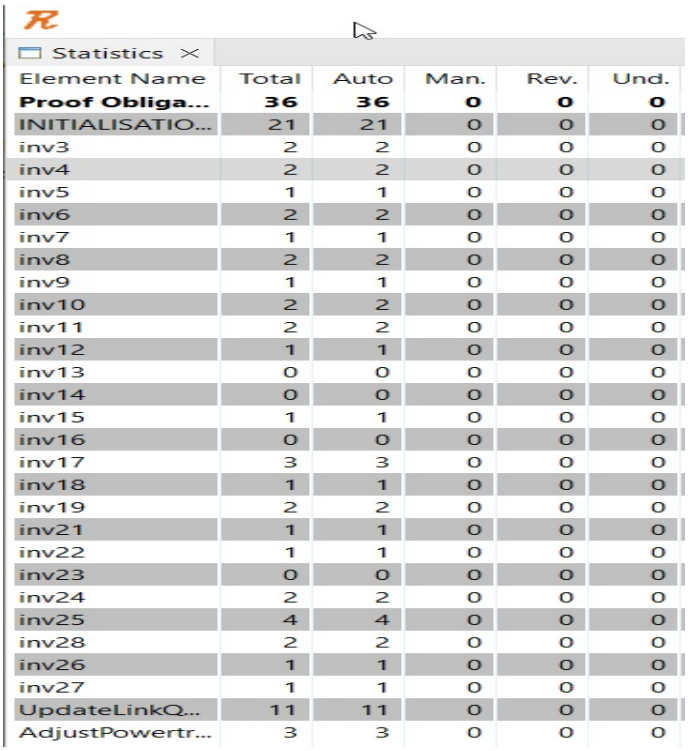


Fig. III.12 – *Rafinement Machine.*

The summary of the proof statistics is indicated in Fig.III.13; around 30 percent of the proof obligations appear in the events and invariant 25 in machine *DataRate*.



Statistics

Element Name	Total	Auto	Man.	Rev.	Und.
Proof Obliga...	**36**	**36**	**0**	**0**	**0**
INITIALISATIO...	21	21	0	0	0
inv3	2	2	0	0	0
inv4	2	2	0	0	0
inv5	1	1	0	0	0
inv6	2	2	0	0	0
inv7	1	1	0	0	0
inv8	2	2	0	0	0
inv9	1	1	0	0	0
inv10	2	2	0	0	0
inv11	2	2	0	0	0
inv12	1	1	0	0	0
inv13	0	0	0	0	0
inv14	0	0	0	0	0
inv15	1	1	0	0	0
inv16	0	0	0	0	0
inv17	3	3	0	0	0
inv18	1	1	0	0	0
inv19	2	2	0	0	0
inv21	1	1	0	0	0
inv22	1	1	0	0	0
inv23	0	0	0	0	0
inv24	2	2	0	0	0
inv25	4	4	0	0	0
inv28	2	2	0	0	0
inv26	1	1	0	0	0
inv27	1	1	0	0	0
UpdateLinkQ...	11	11	0	0	0
AdjustPowertr...	3	3	0	0	0

Fig. III.13 – *ADR Proof Statistics.*

III.6 Conclusion

The selection of appropriate communication parameters can lead to better network performance. The first part focuses on researchers' literature work to improve the spreading factor allocation schemes and presents the effect of the Spread Factor on the performance of LoraWan using the *NS3* simulator. The performances are analyzed in terms of different metrics (delay,

packet delivery ratio, radius, and number of nodes). The simulation results indicate that the message sent at $SF12$ will go further than that sent at $SF7$. Messages with a higher spreading factor will be longer. Indeed, the time in the air longer, and therefore a higher risk of collision.

The second part, we used the formal method to verify and validate the ADR protocol server side in the LoraWan network. We used *Event-B* (the formal method) to model the protocol layers and their properties and Event-B invariants to ensure the protocol's consistency. The event-b provides more rigorous protocol verification and comprehension. By modeling and verifying its properties, we will add more guarantees to the protocol's validity.

The last chapter present the Lora packet format by discussing on the two end device activation methods: Over The Air Activation (OTAA) and Activation By Personalization (ABP) in all versions of LoraWan, and the encryption method using AES cryptography. Our last contribution is to present the deployment of the $AES - 128$ bit encryption / decryption protocol under the $NS - 3$ simulator.

Security and AES-128 Cryptography in LoraWan

IV

Contents

In this chapter, we start with the security goals in the IoT followed by a detailed presentation of AES cryptography, subsequently the authentication of end devices for the two mode ($OTAA$ & ABP), and the encryption and decryption schema will be detailed for all LoraWan versions, our contribution for deploying the $AES-128$ bits algorithm under $NS-3$ simulator and the results obtained will be presented. Finally the security risk and challenges in the IoT are described.

IV.1 Introduction

IoT solutions find LPWAN technology highly attractive, however there is a risk of attacks. Despite the security characteristics of LoRaWAN, security attacks can still affect LoRa devices. For instance, LoRa modulation takes between 900 milliseconds and 1.2 seconds for every LoRa transmission. Because the transmission window is so wide, attackers have many alternatives.

Because the network's end devices have limited resources (battery, computer memory), it is unable to employ cutting-edge cryptographic techniques that need a lot of processing power, making LPWAN security a significant challenge. In order to encrypt messages and decode them using $Appkey$, LPWAN technologies employ the $AES128$ symmetric encryption technique, which creates two keys. Nevertheless, the message remains consistent in length regardless of encryption and decryption, which may facilitate an attacker's ability to decipher the key. Device and network key compromise, replay attacks, and jammer attacks can still affect LPWAN devices in spite of this approach. The microcontroller's radio modules also don't support cryptographic techniques, which makes it difficult to tell if an attacker or the microcontroller is sending the commands [10].

LoRaWAN as a LPWAN protocol ensures low-power, low-cost, long-distance and secure communication for various IoT applications. Security is a fundamental requirement for many IoT applications. The LoRaWAN security policy, which accords with state-of-the-art principles such as the use of a standard security algorithm and end-to-end secure communication protocols, enables mutual authentication, confidentiality and integrity. Therefore, the standardized AES cryptographic algorithm approved by the NIST (National Institute of Standards and Technology) is employed as the LoRaWAN security mechanism. These are generally accepted as one of the best security approaches for network communication encryption. It combines the original AES encryption/decryption algorithm with several modes of operation, including:

- CMAC (Cipher-based Message Authentication Code).
- Counter Mode (CTR).

The former is used to protect the integrity of messages, while the latter is employed for data encryption and decryption. When a new device joins, a unique 128-bit $AppKey$ and a globally unique identifier $DevEUI$ are utilized to generate application session key $AppSKey$ and network session keys ($NwkSKey$ for R1.0, and $FNwkSIntKey$, $SNwkSIntKey$, $NwKSEncKey$ for R1.1).

These session keys can be Activated By Personalisation (ABP) on the production line or during commissioning. It can also be Over-The-Air Activated $(OTAA)$ in the field. Fig.IV.1 show how LoRaWAN traffic is protected using a Network Session key ($NwkSKey$) and an Application Session Key ($AppSKey$). Each payload encrypted by using an $AES-CTR$ algorithm with an AppSKey carries a frame counter to protect the underlying system from replay attacks. A Message Integrity Code (MIC) computed by the AES-$CMAC$ algorithm using the NwkSKey is verified by a network server to ensure packet integrity. When the content or header of a packet is compromised, and the network server cannot compute the correct MIC, the packet is dropped. Double-mode AES algorithms satisfy the security requirements in authentication and packet integrity for end nodes, network servers and application servers.

In the following chapter, the security risk and challenges to the IoT are described in more detail. For this, we will start with the security goals in general, then we will detail the AES standard in Lorawan, the MAC layer and the structure of the Lora packet, then the two end device activation modes (ABP mode and $OTAA$ mode) will be presented. Finally, in all the literature, currently to our knowledge, cryptography using a network simulator was not yet a research topic. Our contribution consists to implement the AES algorithm (encryption and decryption) under the $NS-3$ simulator, and consequently which allows us examine the performances of the LoRa network such as PDR, ToA, Energy and CPU Time.

IV.2 Lorawan Background Security

LoRaWAN defines the networking protocol for LoRa based devices. It specifies different device types, different keys, and encryption capabilities to build a secure wireless network.

LoRaWAN specifies a number of identifiers for devices. All end-devices have a 64-bit unique identifier called Device Identifier $(DevEUI)$ which is set by vendors or developers. In addition, all communication is done using 32 bit device address. Another identifier, called Application Identifier $(AppEUI)$, uniquely identifies the application provider of the end-device.

In order to provide several levels of encryption in LoRaWAN, the cryptographic security is handled by *AES-128* operating in CTR mode [3]. For network and application level packet security, LoRaWAN employs distinct device, network, and application keys. This keeps application data private while enabling intermediary nodes, such gateways and cloud routers, to handle network maintenance and routing duties.

Network Session Key $(NwkSKey)$ and Application Session Keys $(AppSKey)$ are the two session keys that are generated using a $128-bit$ AES key called the Application Key $(App-key)$. To produce and validate the message integrity code (MIC), the network server and the end device share the $NwkSkey$. In addition to ensuring message integrity, this generates a unique signature for each device. Similar to the $NwkSkey$, the $AppSKey$ is used to encrypt and decode the payload of application data. LoRaWAN generates a key stream using $NwkSKey$, $AppSKey$, and the up-link or down-link counter of the messages. As a consequence, each message is encrypted by creating the encrypted payload using the XOR operation with the matching key from the key stream.

The LoRaWAN protocol ensures the security, and provides encryption capability between the end-device and the gateway. However, the payload length is always the same before and after the encryption. This can be used together with overflowing counters by a malicious entity to restore the keystream from the encrypted messages.

• **Security Keys:** The security keys specified in LoRaWAN version 1.1 are shown in Fig.IV.1. It should be noted that the protocol employs two layers of security, one for the network and one for the application. Network encryption ensures the authenticity of the end device in the network, while the application payload encryption ensures authenticity and integrity.

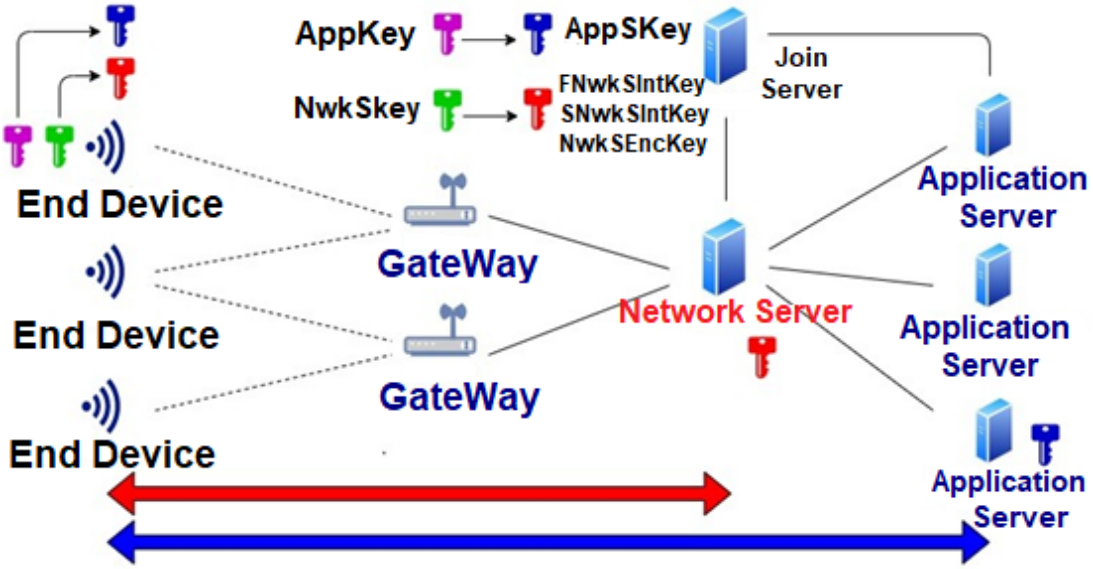


Fig. IV.1 – *Security Keys.*

IV.3 AES Overview

In 1997, the National Institute of Standards and Technology $NIST$ renewed its call to the public cryptography community for the successor to DES. Two Dutch cryptographers – ***Joan Daemen*** and ***Vincent Rijmen*** – submitted the eventual pick known as ***Rijndael***. By 2001, the $NIST$ dubbed it the Advanced Encryption Standard (AES) and officially replaced the use of DES and considered among the strongest encryption algorithm currently in use. AES adopts symmetric block cipher scheme and supports the encryption/decryption key length in 128, 192, and 256 bits [102], these key lengths determine the numbers of rounds in the encryption process to meet different environmental needs.

Table IV.1 – *Different types of AES.*

Parameters	AES-128	AES-192	AES-256
Rounds (Nr)	10	12	14
Key sizes (bits)	128	192	256
Data block size (bits)	128	128	128

Depending on the key length, AES performs 10 to 14 encryption/decryption rounds and it is given in Table IV.1, Similar to other types of encryption, AES algorithm has two inputs: plaintext (data to be encrypted) and cipher key. Plaintext is processed with cipher key through a series of AES encryption algorithms so that it becomes ciphertext. Cipher key is generated from key generation process. Each of which (except the last round) consists of four steps, i.e., $SubBytes, ShiftRows, MixColumns$, and $AddRoundKey$, which are briefly described in the following:

• **SubBytes:** a nonlinear and invertible transformation. SubBytes often prepares a substitution table (i.e., SBox) or combinational logic to individually map bytes of a data array into other bytes. SBox entries are produced by calculating multiplicative inverses in a Galois Field by using an affine transformation.

• **ShiftRows:** byte transposition by rotating rows of the data array according to predefined offsets.

• **MixColumns:** multiplying each column of the data array by a modular polynomial equation or prime number in a Galois Field. Instead of computing separately, MixColumns can also be implemented by using large Lookup Tables.

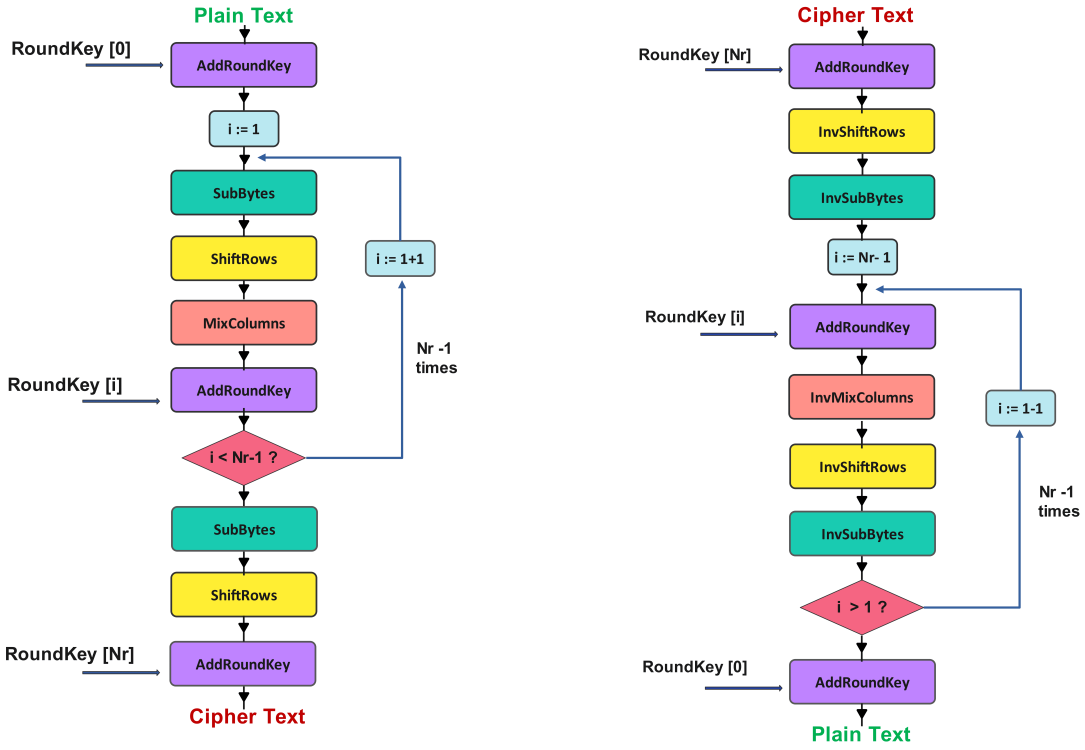
• **AddRoundKey:** adding the data array with round-keys that are derived from an initial encryption key in the key expansion unit. This function XOR's each byte of the data block with the corresponding byte in the round-key.

A flowchart describing AES encryption in terms of basic operations — SubBytes, ShiftRows, MixColumns, and AddRoundKey — is shown in Fig.IV.2a note that the number of cipher rounds, Nr, depends on the size of an encryption key. The last round, number Nr, is slightly different from the remaining Nr - 1 rounds, in that it does not contain the MixColumns operation.

By a straightforward inversion of the order of operations and by replacing all basic operations by their respective inverses, we obtain the AES decryption flowchart shown in Fig.IV.2b.

IV.3.1 Security requirements:

In order to discuss the security in LoRaWAN, the LoRa-MAC layer format must first be described. Further, the security requirements of IoT applications are explained:



(a) *AES Encryption flowchart.* (b) *AES Decryption flowchart.*

Fig. IV.2 – *AES Encryption / Decryption flowchart*

1. Confidentiality: safeguarding the data by keeping private information out of the hands of unauthorized users.

2. Integrity: avoiding the alteration of data being transferred between IoT devices by unauthorized users.

•Non-repudiation: This means that the source of the data cannot deny or repudiate that the data originates from the sender, ensures non-repudiation by means of digital signatures

3. Availability: guaranteeing that information and resources are accessible when authorized users seek them. The availability of IoT devices is compromised by new physical layer assaults (such as the depletion-of-battery attack).

4. Authenticity: Verifying the parties involved in the transmission helps to ensure that the data being conveyed is legitimate.

Typically, the attacker goals are:
1. compromising the network security properties (i.e. availability, confidentiality, authenticity, etc.).
2. compromising the network security assets (i.e. keys, nonces, counters, etc.).

It is crucial to remember that LoRaWAN's widespread adoption is impacted by the requirement to secure its component parts.

IV.3.1.1 Analysis of Security Requirements in LoraWan

Security is probably the main concern in any IoT application. Several Confidentiality, Integrity, and Availability (CIA) enforcing mechanisms are described in LoRaWAN specifications [3]. The LoRaWAN CIA mechanisms exploit the AES encrypting algorithm and

128 bit keys. End-devices (univocally identified by a IEEE EUI64 address, the DevEUI) can use two different procedures to derive and distribute the encryption keys for joining a LoRaWAN: the Over-the-Air Activation (OTAA), and the Activation By Personalization (ABP). In OTAA two root keys are hardwired in the node, which are used, together with information obtained during the join procedure, to derive actual session keys. On the contrary, in the ABP mode all the relevant information are stored in the node, which does not need a Join procedure. Confidentiality and integrity of MAC commands is done at the network level and user data confidentiality is implemented at the application level. Starting from the latest specification release, additional join and network session keys have been defined as well for increasing security and flexibility; in order to comply with the following security requirements:

• Authentication
The authentication requirement is guaranteed through the use of secret keys by the IoT devices towards the server. When a device computes the MIC value of the MAC packet, in addition to preventing the modification of the packet, it provides proof of authentication: only a device that owns the session key will be able to produce this result.

• Confidentiality
To ensure confidentiality, LoRaWAN implements the AES data encryption mechanism with 128-bit keys operating in counter mode (AES-128-CTR) and the secret key AppSKey. AppKey and NwkKey keys never change, representing a potential vulnerability.

• Integrity
As regards integrity, the LoRaWAN protocol natively satisfies such a requirement, without the need to introduce any additional external mechanisms. LoRaWAN, before transmitting a message, generates the MIC value, using the $NwkSKey$ and the $AES - CMAC$ algorithm. The calculation of this value takes into account the entire header of the MAC packet and its payload. The goal of the MIC value is to avoid receiving tampered information from malicious entities.

• Authorization
Concerning authorization, LoRaWAN guarantees this requirement by introducing two activation methods (i.e., $OTAA$ and ABP). With these two mechanisms, a device is authorized to communicate securely with a network server.

IV.4 LoRa Physical Packet Structure

Packet format in LoRa includes the following fields: starts with a preamble which is used for the synchronization between the receiver and the transmitter, payload, and payload Cyclic Redundancy Check (CRC). Additionally, packets can take one of the two following forms: explicit and implicit. As depicted in Fig.IV.3, the difference lies in the fact that first form packet format contains a header and a CRC for the header, in order to verify the integrity of the packet. It is noted that the header is always encoded with a $CR = \frac{4}{8}$. The payload is encoded with a variable CR.

Payload is a variable field, which contains the data from end-devices. The header field has 2 bytes and gives information related to the payload length, the CR as well as the presence or not of CRC payload. Of note, the payload error detection is performed only in uplink traffic. On the other hand, in implicit format header is not necessary, as both sides have set CR and the presence of CRC before initiation of messaging. Following this procedure, the transmission time can be reduced in comparison to the explicit mode [3].

Downlink and uplink packets are the two types of packets used in LoRa. The devices trans-

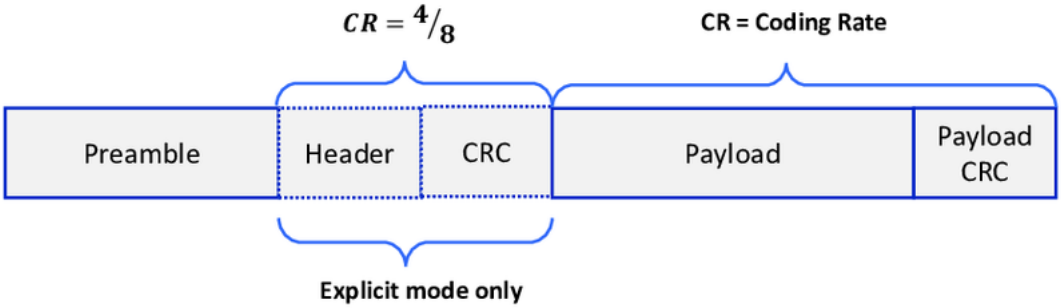


Fig. IV.3 – *LoRa packet format.*

mit uplink data to the NS, which is then relayed via one or more gateways. Uplink packets employ the LoRa radio packet explicit mode, which includes a header CRC(PHDR_CRC) and a LoRa physical header (PHDR, which gives information about the length of the physical layer payload). The CRC payload is present in the LoRa physical frame, but it is only present in UPLINKS. The integrity of the payload is protected by a CRC.

The structure of the LoRaWAN packet is defined by the LoRaWAN specification and is shown in Fig.IV.4 Transmission of LoRa packets is initiated by a preamble, which is followed by the radio layer along with a MAC layer of a packet.

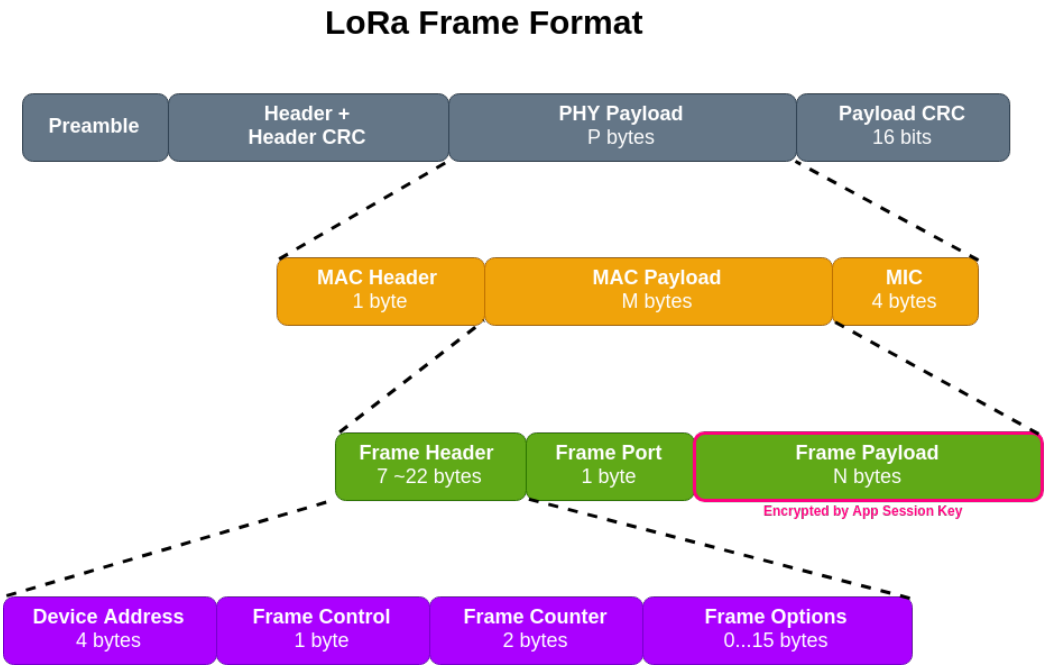


Fig. IV.4 – *Structure of LoRaWAN (Long Range Wide Area Network) packet.*

The preamble header is an essential part of the packet since the receiver filters the incoming traffic from the received preamble. The MAC layer is located above the physical layer and is responsible for encrypting application data. All of the information required to forward the message across the network is contained in the physical payload (PHYPayload) that is carried by both uplink and downlink messages. The physical payload is made up of a MAC payload, a 4-byte message integrity code (MIC) that uses the AES-128 protocol to guarantee the integrity of the packet, and a single-byte MAC header (MHDR) that specifies the message type, metadata, and protocol version of the message [103].

The Frame Header (FHDR), the Port field (FPort), which provides the device address, and

the optional MAC Frame Payload (FRMPayload) are the three core elements of the MAC Payload. MAC payload contains encrypted information.

The frame header (FHDR) is the data link layer packet header found inside the MAC payload. Device address (DevAddr), frame control (FCtrl), frame counter (FCnt), and up to 15 bytes of frame options (FOpts) for carrying MAC commands make up the frame header. This type of counter, called FCnt, enables the devices to record the quantity of data frames sent uplink to the Network Server (FCntUp) and downlink to the device from the Network Server (FCntDown). Whenever an OTAA end-device successfully process a join procedure their frame counters is reset to 0 while the ABP devices have theirFCnt initialized to 0 at fabrication ant it must never be reset to 0. Frame Counter (FCnt) in LoRaWAN is also used to prevent possible replay attacks.

IV.4.1 MAC Message Types

LoRaWAN foresees different "types" of packets. The information characterizing the type of a packet is encoded with three bits in the field *Mtype* in the MAC Header $(MHDR)$, which is sent as clear-text with the packet and it is reported inside the dataset [26]. All the possible *Mtype* values with their description is reported in Table IV.2.

Table IV.2 – *LoRaWAN MAC message types*

MAC message type	Description
000	Join Request
001	Join Accept
010	Unconfirmed Data Up
011	Unconfirmed Data Down
100	Confirmed Data Up
101	Confirmed Data Down
110	RFU / Rejoin-request
111	Proprietary

- **Join Request & Join Accept:** These messages are used to establish connection between LoRa end device and Gateway; and are used by the Over The Air Activation (OTAA) process. Data messages can be either MAC Commands or Data messages.
- **Confirmed Data Message:** This message type requires to be acknowledged by its receiver.
- **Unconfirmed Data Message:** This message type does not require any acknowledgment.
- **Proprietary:** This message type is used to incorporate non standard message format functionalities.
- **RFU:** LoRaWAN 1.0.x - Reserved for Future Usage
- **Rejoin-request:** LoRaWAN 1.1 - Uplink over-the-air activation (OTAA) Rejoin-request

IV.4.2 LoRaWAN MAC Commands

MAC commands are used for network administration between server and end device. These commands are non visible to applications running in the LoRa server and end devices. A single data frame consists of one or multiple MAC commands(either piggybacked or transmitted as separate frame). MAC commands are segregated based on CID field of size 1 octet long. CID stands for Command IDentifier. These mac commands are used by end device or by gateway or by both.

Following Table IV.3 mentions list of *LoRaWANMAC* commands with *CID*, transmitted by end device or gateway and their functions [26], each having a request command and an answer command.

Table IV.3 – *LoRaWAN MAC Commands.*

CID	Command	Transmitted By	Description
0x02	LinkCheckReq	Node	Connection validation.
0x03	LinkCheckAns	GW	Reply to LinkCheckReq.
0x03	LinkADRReq	GW	Adjust node DR,tx power,rx rate or channel.
0x04	LinkADRAns	Node	Response to LinkADRReq.
0x04	DutyCycleReq	GW	Set tx duty cycle.
0x04	DutyCycleAns	Node	Response DutyCycleReq.
0x05	RXParamSetupReq	GW	Set rx slot parameters.
0x05	RXParamSetupAns	Node	Response RXParamSetupReq.
0x06	DevStatusReq	GW	Request status of node.
0x06	DevStatusAns	Node	Response to DevStatusReq with battery level ad demodulation margin.
0x07	NewChannelReq	GW	Setup/modify new radio channel.
0x07	NewChannelAns	Node	Response to NewChannelReq.
0x08	RXTimingSetupReq	GW	Set reception slot timing.
0x08	RXTimingSetupAns	Node	Response to RXTimingSetupReq.
0x08-0xFF	Proprietary	Node/GW	Reserved further extensions.

LinkCheckReq command is used by end-device to validate network connectivity.

LinkCheckAns The GW responds with GwCnt, which shows how many GWs successfully received the request.

LinkADRReq is transmitted across the nodes from GW in order to adopt a data rate. DataRate and Tx output power (TXPower) with region-specific values are contained in the ADR (Adaptive Data Rate) message. Additionally, the message's ChMask field is used to encode suitable uplink channels.

LinkADRAns is transmitted by end devices to indicate whether the required channel states, Tx Power, and DataRate are successfully implemented or discarded.

Aggregated maximum (Equation IV.1) Duty Cycle can be managed for all sub-bands by using the *DutyCycleReq* request and message, which includes a *MaxDCycle* parameter in the range [0:15]. The remote device can be turned off instantly by pressing the 255 off switch, whereas 0 indicates that there is no duty cycle limits.

$$Aggreaged\ duty\ cycle = \frac{1}{2^{MaxDCycle}} \tag{IV.1}$$

DutyCycleAns is a reply to *DutyCycleReq* without any payload.

RXParamSetupReq controls the second receive window (RX2) data rate and the frequency for each uplink.

RXParamSetupAns contains status bits indicating that configuration is successful or invalid.

DevStatusReq is a blank message that's used to see how end devices are doing. The message is returned by the end-device with the battery level and the modulation margin of the SNR ratio using *DevStatusAns*.

The Network Server can use the *NewChannelReq* request to add a new channel or change an existing one. The central frequency and range of viable data rates are contained in the

message. The protocol states that end devices must manage 16 distinct channels, indexed from 0 to N −1, where N is the number of default channels. With 100 Hz increments, the frequency can be adjusted between 100 and 1670 MHz, with the lower 100 MHz held aside for future use cases. Channel disabling occurs when the frequency value is set to 0. Using four bit indices, the DrRange subfield in *NewChannelReq* defines the allowed data range for the specified channel. $MinDR, MaxDR$ are the min and max data rates for the *DrRange* field.

End-device sends an ACK frame indicating the status of the *NewChannelReq* request either successful or not.

The delay between uplink Tx and the first reception window is configured with *RXTimingSetupReq* message. The second reception window is opened after one (01) second of the first window.

IV.5 End Device Activation

LoRaWAN uses modern encryption scheme, i.e., Advanced Encryption Standard (AES) [103], to guarantee its end-to-end security. The basic features include bi-directional authentication, integrity checking, and data encryption. Bi-directional authentication between end-devices and network servers ensures that only authenticated devices can be connected to LoRaWAN. If a node is not activated, the server will ignore its messages. A node is considered active when it has a valid copy of the following values: (i) Device Address (DevAddr), which consists of the device address; (ii) Network Session Key (NwkSKey), or the key used to compute the integrity value of MAC packets; (iii) Application Session Key (AppSKey), which indicates the key used to encrypt messages.

There are two activation methods available:

- Over-The-Air-Activation (OTAA): the safest and most suggested way to activate end devices. The network and devices go through a join process where security keys are negotiated and a dynamic device address is assigned.
- Activation By Personalization (ABP): demands that both the device's security keys and address be hardcoded. ABP has the disadvantage of not allowing devices to swap network providers without manually changing the device's keys, making it less secure than OTAA.

The join procedure for LoRaWAN 1.0.x and 1.1 is slightly different. The following two sections describe the join procedure for LoRaWAN 1.0.x and 1.1 separately.

IV.5.1 Over The Air Activation (OTAA) in LoRaWAN 1.0.x

For an end-device to join a LoRaWAN network, one of the procedures should be followed. Over-the-Air Activation $(OTAA)$ requires $DevEUI$, the $AppEUI$, and an $AppKey$. This process must be followed each time an end device joins a new network or loses the session key data. Since an end-device-specific network session key is created each time the device connects to the network, $OTAA$ is referred to as the most secure authentication method. This permits roaming between several carriers' networks. Additionally, even if one of the keys is compromised, having two keys makes it more difficult to view or tamper with application data. The $OTAA$ process is initiated by the end-device through the transmission of a join-request message. The message contains the end device's $AppEUI$, $DevEUI$, and nonce $(DevNonce)$. The network server tracks a random value called $DevNonce$, which is used to deny any join request that contains an invalid nonce value. Replay attacks are stopped by this method.

In LoRaWAN 1.0.x, the join procedure requires two MAC messages to be exchanged between the end device and the Network Server:

- **Join-request:** from end device to the Network Server.
- **Join-accept:** from Network Server to the end device.

Prior to activation, the end device must contain the $AppEUI$, $DevEUI$, and $AppKey$. The root key is an AES-128 bit secret key called $AppKey$. It is necessary to provision the same $AppKey$ on the network where the final device will register. Both the $AppEUI$ and the $DevEUI$ are publicly viewable and not confidential.

➢ The ***AppKey*** is never sent over the network.

The following steps as shown in Fig.IV.5 describe the Over-The-Air-Activation (OTAA) procedure.

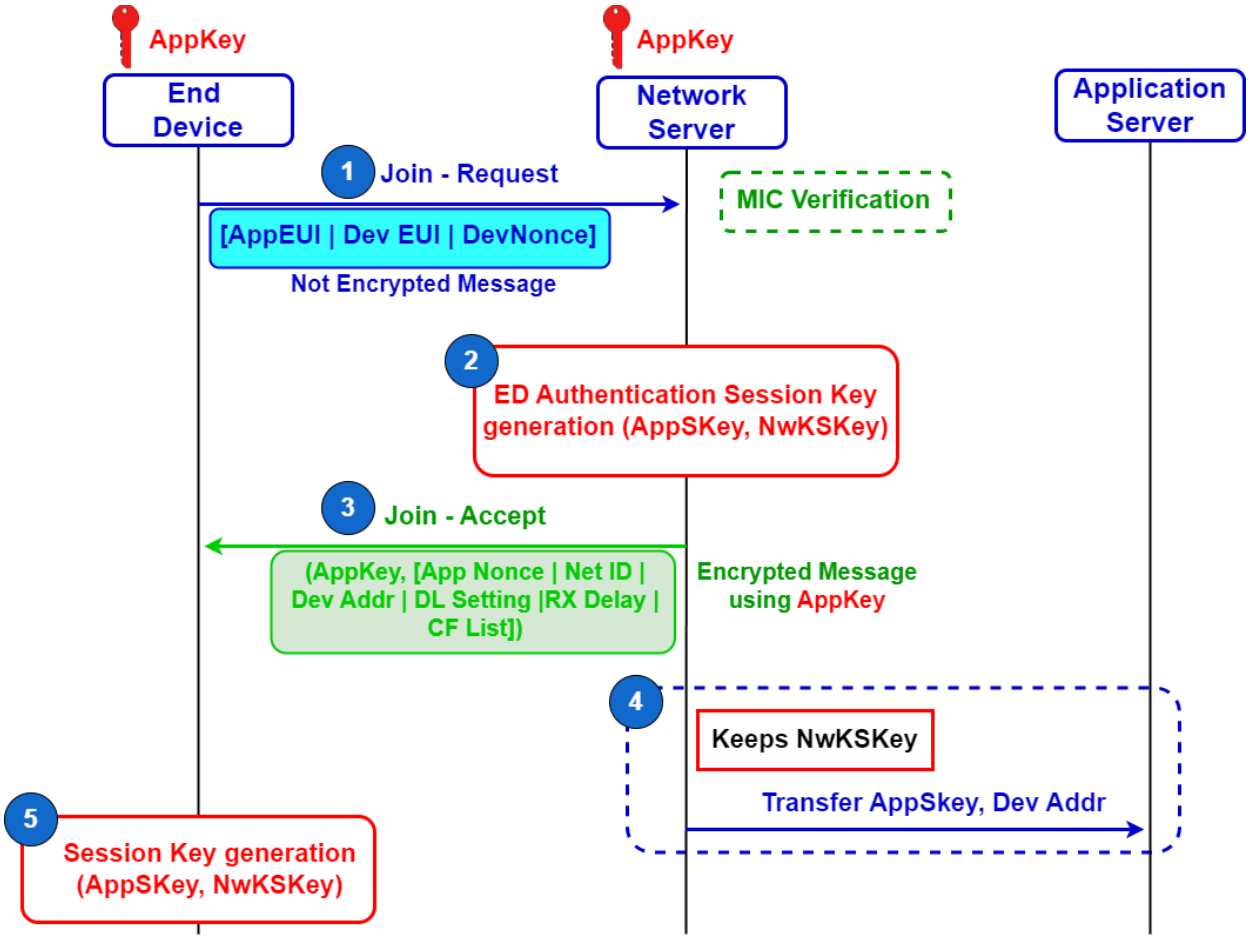


Fig. IV.5 – *OTAA message flow in LoRaWAN 1.0.*

Step1: The join procedure is always started by the end device. The end device transmits the $Join-request$ message to join the network. This $Join-request$ message has the following fields:

- **AppEUI :** a 64-bit globally unique application identifier in IEEE EUI64 address space that uniquely identifies the entity able to process the Join-req frame.
- **DevEUI :** a 64-bit globally unique device identifier in IEEE EUI64 address space that uniquely identifies the end-device.
- **DevNonce:** A 2-byte value that is unique, random, and produced by the end device. The Network Server monitors each end-device's join request using its DevNonce. When

an end device makes a Join-request with a previously used DevNonce (referred to as a replay attack[104]), the Network Server denies the request and prevents the end device from registering on the network.

The **Message Integrity Code (MIC)** is calculated over all the fields in the Join-request message using the ***AppKey***.

A MIC value (Message integrity code) is calculated by the formula (IV.2) and (IV.3).

$$cmac = AES128\_cmac(AppKey, MHDR|AppEUI|DevEUI|DevNonce) \quad \text{(IV.2)}$$

$$MIC = cmac[0\ldots3]. \quad \text{(IV.3)}$$

The calculated MIC is then added to the $Join-request$ message.

The $AppKey$ is not sent with the $Join-request$ message, and the $Join-request$ message is not encrypted.

The $Join-Request$ message format is shown in Fig.IV.6.

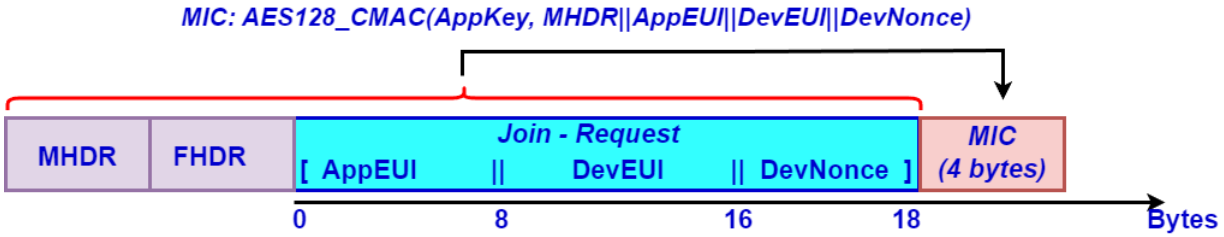


Fig. IV.6 – *The Join-Request message structure.*

Step2: The Join-request message is processed by the Network Server. If the end device is allowed to join a network, the Network Server will create two session keys ($NwkSKey$ and $AppSKey$) along with the Join-accept message.

The Join-accept message consists of the following fields.

- **AppNonce :** a unique ID or random value that the network server provides. The end device derives the two session keys, $AppSKey$ and $NwkSKey$, from the AppNonce.
- **NetID :** consists the most important seven bits of the network identification (NwkID).
- **DevAddr :** To identify the end device within the existing network, the Network Server assigns a 32-bit device address.
- **DLSettings :** Downlink parameters that the end device should utilize are contained in a 1-byte field.
- **RxDelay :** contains the delay between TX and RX.
- **CFList :** an optional list of the network's channel frequencies that the end device is joining. These are regionally specific frequencies.

With the AppKey, the Message Integrity Code (MIC) is computed across all the fields in the Join-accept message. Following that, the Join-accept message is updated with the computed MIC. The following is how the network server determines the NwkSKey and AppSKey after confirming the join-request message:

$$NwkSKey = AES128\_Encrypt(AppKey, 0x01|AppNonce|NetID|DevNonce|pad\_16). \quad \text{(IV.4)}$$

$$AppSKey = AES128\_Encrypt(AppKey, 0x02|AppNonce|NetID|DevNonce|pad\_16). \quad \text{(IV.5)}$$

where 0x01 and 0x02 are the port fields for specific applications and pad 16 is a function adding zero octets to make the length of the data a multiple of sixteen. AppKey encrypts the join-accept message as follows:

$$Join-accept = AES128\_Encrypt(AppNonce|NetID|DevAddr|RFU|RxDelay|CFList|MIC. \tag{IV.6}$$

Following that, the $AppKey$ is used to encrypt the Join-accept message itself. The Network Server encrypts the $Join-accept$ message using a AES decrypt operation in ECB mode.

Step3: The Network Server uses a standard downlink to send the encrypted Join-accept message back to the end device. If the Network Server does not accept the $Join-request$ message, the end-device does not receive any response.

The $Join-Accept$ message structure is shown in Fig.IV.7.

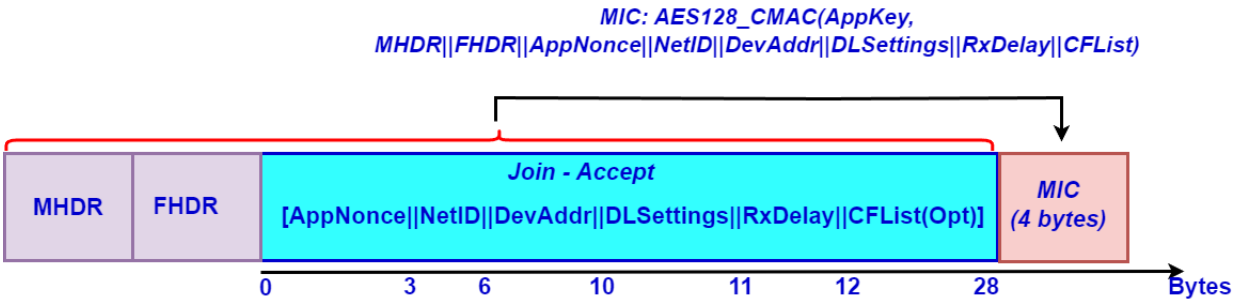


Fig. IV.7 – *The structure of Join-Accept messages.*

Step4: The Network Server keeps the $NwkSKey$ and distributes the $AppSKey$ to the Application Server.

Step5: The end device utilizes the AES encryption process to decipher the $Join-accept$ message. Using the $AppKey$ and $AppNonce$, the end device obtains the two session keys: the Application Session Key ($AppSKey$) and the Network Session Key ($NwkSKey$).

After activation, the following additional information is stored in the end device.

- **DevAddr :** a 32-bit hexadecimal number, device address assigned by the Network Server to identify the end device within the current network, which can be divided into two parts: the network identifier and network address.
  - **Network identifier (NwkID):** The NwkID is a 7- bit network identifier used to distinguish between the addresses of overlapping networks operated by different network operators.
  - **Network address (NwkAddr):** The NwkAddr is a 25-bit network address assigned by the network manager.
- **NwkSKey :** To ensure message integrity, the end device and network server employ the network session key to determine and validate each data message's Message Integrity Code (MIC). Along with authentication, the $NwkSKey$ is utilized to encrypt and decrypt payloads using MAC instructions.
- **AppSKey :** To ensure message confidentiality, application payloads in data messages are encrypted and decrypted using the application session key.

IV.5.2 Over The Air Activation (OTAA) in LoRaWAN 1.1

The end device should store the JoinEUI, DevEUI, AppKey, and NwkKey prior to activation. The *AppKey* and *NwkKey* are root keys, which are *AES* − 128 bit secret keys. In order to facilitate the processing of the join procedure and session key derivation, the corresponding *AppKey*, *NwkKey*, and *DevEUI* ought to be provisioned onto the Join Server. The *JoinEUI* and *DevEUI* are open to the public and not confidential.

➢ The ***AppKey*** and ***NwkKey*** are never sent over the network.

The following steps showing in Fig.IV.8 describe the Over-The-Air-Activation (*OTAA*) procedure.

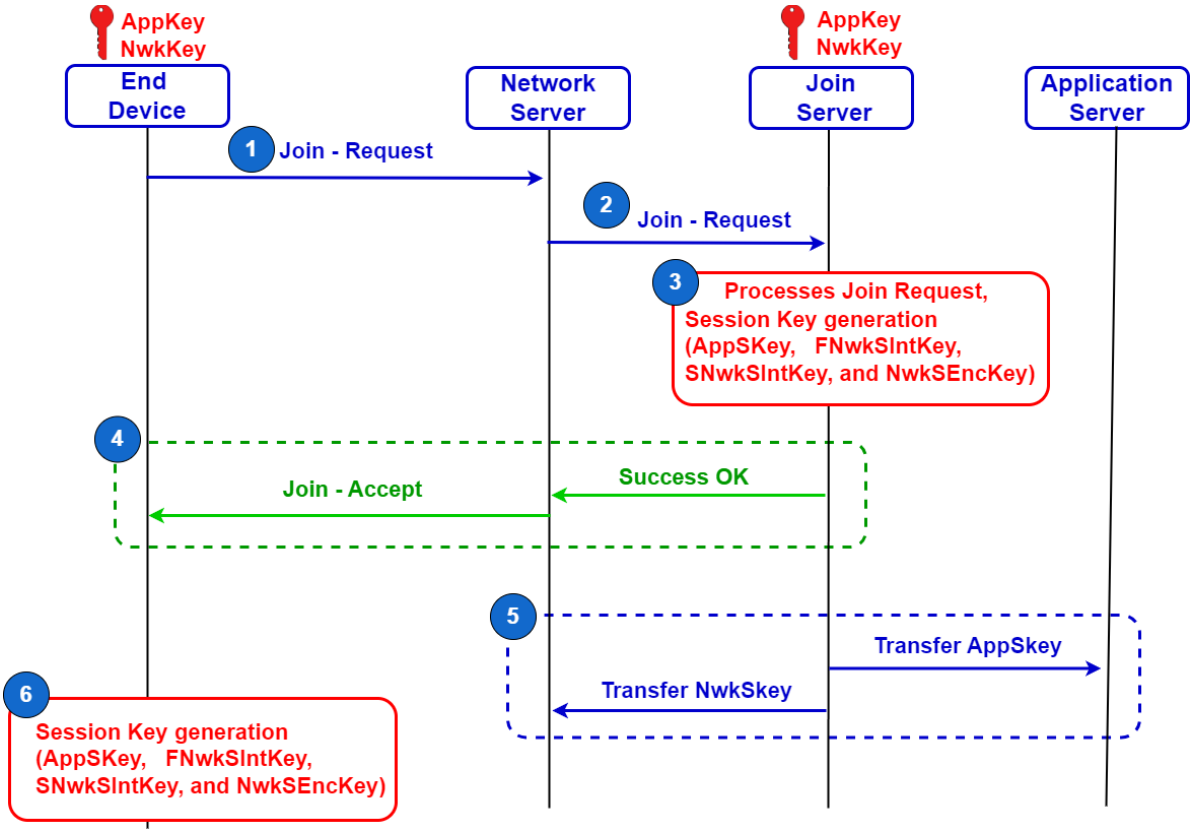


Fig. IV.8 – *OTAA message flow in LoRaWAN 1.1.*

Step1: The join procedure is always initiated by the end device. The end device sends the *Join − request* message to the network that is going to be joined. The *Join − request* message consists of the following fields.

- **JoinEUI :** An IEEE EUI64 address space 64-bit global application identification that uniquely identifies the Join Server capable of executing the *Join − request* and deriving the session keys.
- **DevEUI :** a 64-bit global device identifier in IEEE EUI64 address space that uniquely identifies the end-device.
- **DevNonce :** a 2-byte counter, starting at 0 when the device is initially powered up and incremented with every *Join − request*. The DevNonce value is used to prevent replay attacks[105] since it cannot be used twice for a given *JoinEUI*.

In LoRaWAN 1.1 $AppEUI$ is replaced with the $JoinEUI$.

The MIC is calculated over all the fields in the $Join-request$ message using the $NwkKey$. The calculated MIC is then added to the $Join-request$ message. The following formula calculates the MIC code using the AES 128 protocol in the authentication mode:

Using the $NwkKey$, the MIC is computed over all the fields in the $Join-request$ message. The $Join-request$ message is then updated with the determined MIC. The MIC code is computed using the AES 128 protocol in the authentication mode using the formula.IV.7 below:

$$MIC = AES128\_CMAC(NwkKey, MHDR|AppEUI|DevEUI|DevNonce \quad \text{(IV.7)}$$

The NwkKey is not sent with the $Join-request$ message, and the $Join-request$ message is not encrypted but sent as plain text.

The network root key, $NwkKey$, was previously preshared between the network server and the end node (either at commissioning or as set by the makers).

Step2: The Network Server forwards the Join-request message to the corresponding Join Server.

Step3: The Join-request message is processed by the Join Server. If the end-device is allowed to join the network, the Join Server will produce all of the session keys (AppSKey, FNwkSIntKey, SNwkSIntKey, and NwkSEncKey).

Step4: The Network Server creates the Join-accept message if the aforementioned process is successful. The following fields make up the Join-accept message:

- **JoinNonce :** FNwkSIntKey, SNwkSIntKey, NwkSEncKey, and AppSKey are the session keys that the end device uses to determine JoinNonce, a device-specific counter value supplied by the Join Server.
- **NetID :** a 24-bit unique network identifier.
- **DevAddr :** To identify the end device within the existing network, the Network Server assigns a 32-bit device address.
- **DLSettings :** a 1-byte field consisting of downlink settings which the end device should use.
- **RxDelay :** contains delay between TX and RX.
- **CFList :** an optional list of the network's channel frequencies that the end device is joining. These are regionally specific frequencies.

Step5: The three network session keys—FNwkSIntKey, SNwkSIntKey, and NwkSEncKey—are sent to the Network Server by the Join Server, while the AppSKey is sent to the Application Server.

Step6: The Join-accept message is decrypted by the end-device using the AES encryption procedure. The final device generates session keys using AppKey, NwkKey, and JoinNonce.

After activation, the following additional information is stored in the end device.

- **DevAddr :** a 32-bit device address assigned by the Network Server to identify the end device within the current network.

- **FNwkSIntKey :** In order to ensure message integrity, the end device uses the network session integrity key to determine the partial MIC (the second two bytes) of all uplink data messages. The following is a representation of the key derivation:

$$\text{FNwkSIntKey} = \text{AES128\_Encrypt}(\text{NwkKey}, 0x01 \,|\, \text{JoinNonce} \mid \text{JoinEUI} \mid \text{DevNonce} \mid \text{pad}_{16}) \tag{IV.8}$$

- **SNwkSIntKey :** The first two bytes of the MIC of every uplink data packet are determined using the serving network session integrity key, which is likewise a 128-bit network session key. The whole 32-bit MIC of the entire downlink data message is likewise computed using it. To ensure message integrity, the MIC of the downlink data message would be checked in ED. The following is a representation of its key derivation:

$$\text{SNwkSIntKey} = \text{AES128\_Encrypt}(\text{NwkKey}, 0x03 \,|\, \text{JoinNonce} \mid \text{JoinEUI} \mid \text{DevNonce} \mid \text{pad}_{16}) \tag{IV.9}$$

- **NwkSEncKey :** To ensure message secrecy, the network session encryption key is utilized to encrypt and decrypt the payloads of the uplink and downlink data messages using MAC instructions. The only LoRa devices that store this information are NS and ED. The following is its derivation:

$$\text{NwkSEncKey} = \text{AES128\_Encrypt}(\text{NwkKey}, 0x04 \,|\, \text{JoinNonce} \mid \text{JoinEUI} \mid \text{DevNonce} \mid \text{pad}_{16}) \tag{IV.10}$$

- **AppSKey :** To ensure message confidentiality, the application data in the data messages is encrypted and decrypted using a 128-bit application session key that is utilized by the Application Server and the end device. The derivation is as follows:

$$\text{AppSKey} = \text{AES128\_Encrypt}(\text{AppKey}, 0x02 \,|\, \text{JoinNonce} \mid \text{JoinEUI} \mid \text{DevNonce} \mid \text{pad}_{16}) \tag{IV.11}$$

IV.5.3 Activation By Personalization (ABP):

The second joining procedure is Activation by Personalization (ABP). This process connects devices to the designated network directly, without the need to start a join-request and accept process. The disadvantage of activation by personalization is that it is less secure and does not allow devices to swap network providers without manually changing the device's credentials. There is no Join Server in the ABP process.

In the end-device, the device addresses $DevAddr$, $NwkSKey$, and $AppSKey$ are defined and saved directly. It can therefore use these keys to encrypt communications immediately without creating any keys. In fact, session keys remain unchanged throughout the device's life, unless the user modifies them manually or the firmware is updated. All communications between the device, gateway, and network server can be decrypted by outside parties for the duration of the device's life if the keys are compromised.

IV.5.4 Activation By Personalisation in LoRaWAN 1.0.x :

In place of the $DevEUI, AppEUI$, and $AppKey$, the DevAddr and the two session keys $NwkSKey$ and $AppSKey$ are placed directly into the end-device. A distinct set of $NwkSKey$ and

AppSkey should be present on every end device. As illustrated in Fig.IV.9, the same *DevAddr* and *NwkSKey* should be kept in the Network Server, while the *AppSKey* should be kept in the Application Server.

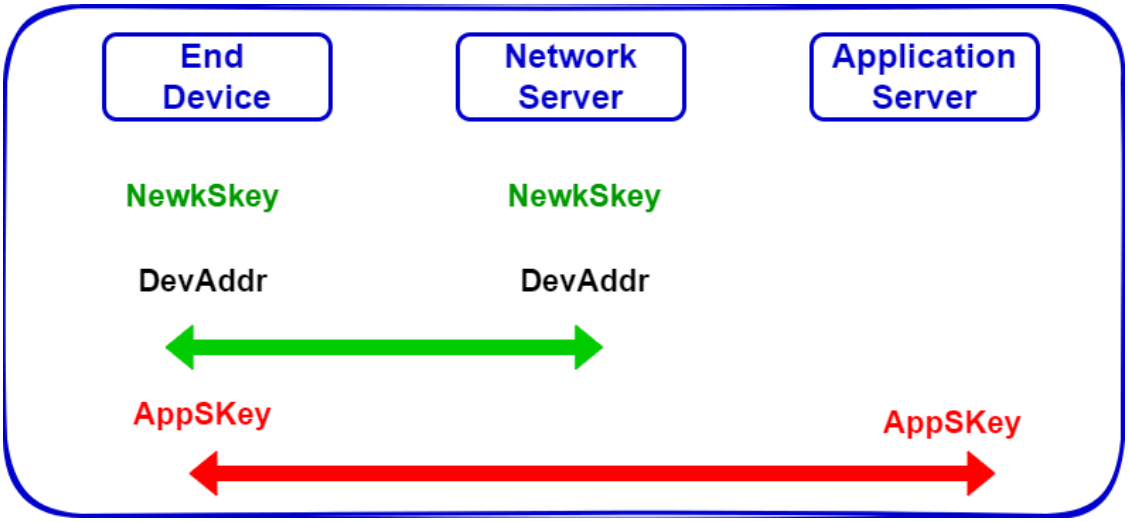


Fig. IV.9 – *Pre-sharing DevAddr and session keys for ABP in LoRaWAN 1.0.*

IV.5.5 Activation By Personalisation in LoRaWAN 1.1 :

The *DevAddr* and the four-session keys *FNwkSIntKey*, *SNwkSIntKey*, *NwkSEncKey*, and *AppSKey* are directly stored into the end device instead of the *DevEUI*, *JoinEUI*, *AppKey*, and *NwkKey*. The same *DevAddr*, *FNwkSIntKey*, *SNwkSIntKey*, and *NwkSEncKey* should be stored in the Network Server and the *AppSKey* should be stored in the Application Server as shown in Fig.IV.10.

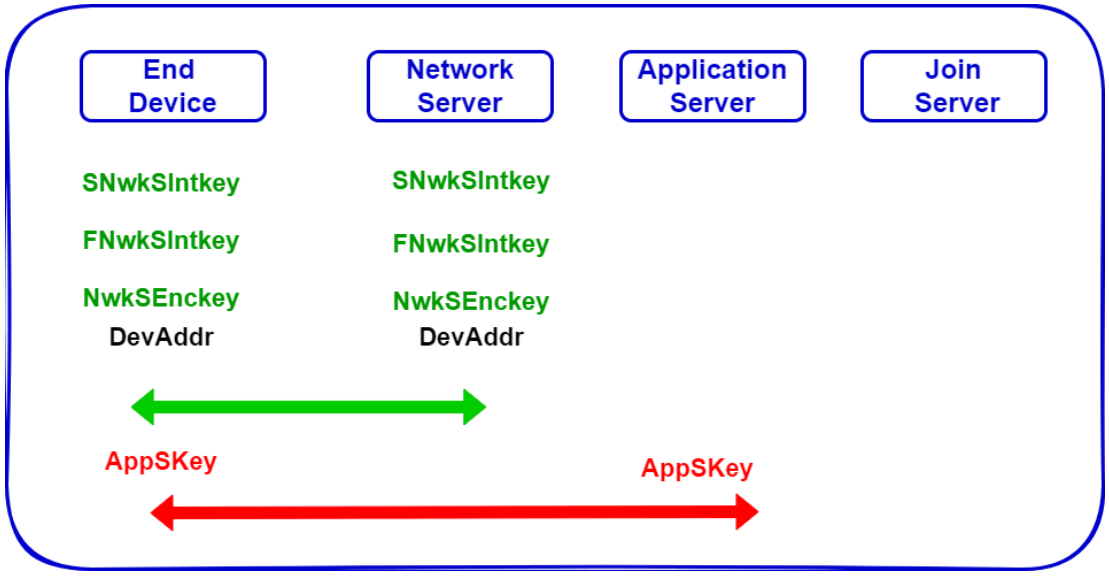


Fig. IV.10 – *Pre-sharing DevAddr and session keys for ABP in LoRaWAN 1.1.*

The key distinctions between *OTAA* and *ABP* activation modes are outlined in Table IV.4, which provides a summary of their differences.

Table IV.4 – *Comparison of LoRaWAN Activation Modes: ABP Vs. OTAA.*

Properties	Activation By Personalization (ABP)	Over-The-Air Activation (OTAA)
Session Keys	End-devices are pre-provisioned with static session keys for their entire operational lifetime	Dynamically generates and periodically renews session keys for end-devices
Network Connection	End-devices are permanently associated with a specific network	End-devices have the flexibility to switch between networks dynamically
Join Procedure	No join procedure required	Join procedure mandatory
Security Level	Basic security due to static keys	Enhanced security through dynamic key generation
Configuration Complexity	Relatively simple to configure	More complex to configure

IV.5.6 Rejoin-request

Rejoin-request handling and configuration for LoRa End Devices are supported by LoRa Technology. By sending a rejoin-request, a device may request the network server to reactivate without being "disconnected" until it receives the activation. That is, the device will continue to use the previous security context if it does not get a fresh activation.

It is possible to restore a lost session context, re-key a device (device address, session-keys, and frame-counters), or reset the device context including all radio parameters (device address, frame counters, session-keys, and radio parameters) using the rejoin-request [3].

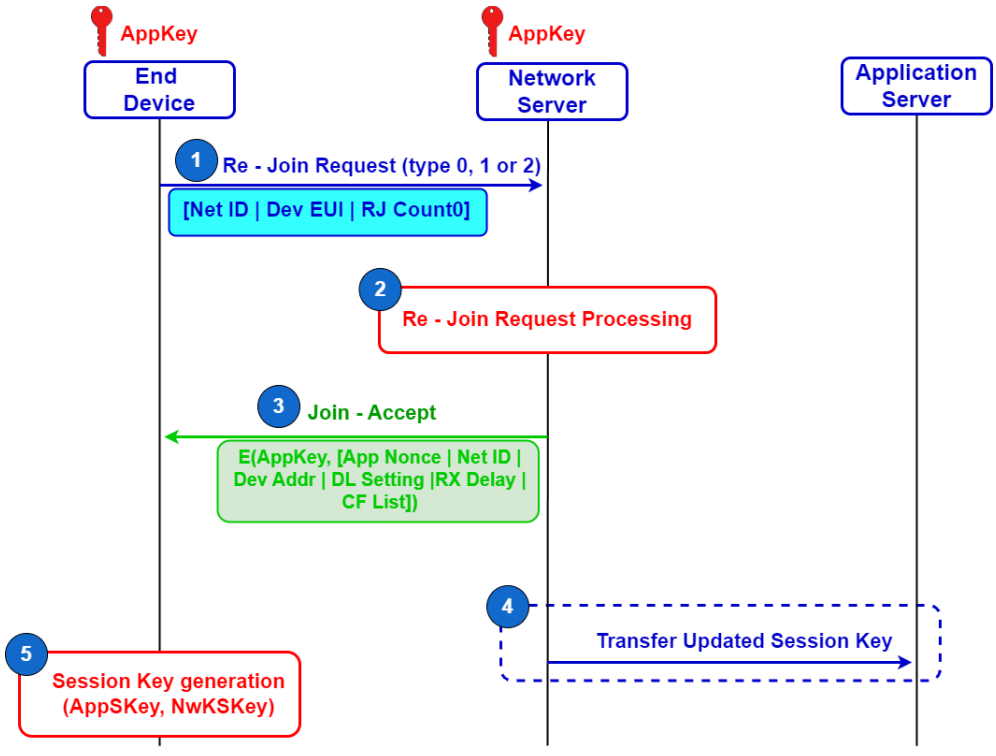


Fig. IV.11 – *Lora Rejoin Procedure.*

IV.5.6.1 Types of Rejoin Requests

Rejoin − request messages come in three different types: Type 0, Type 1, and Type 2. The Rejoin Type is indicated by the first byte of the *RejoinRequest* message. The end device's new session context is initialized using these message types. The network responds to the *Rejoin − request* message with a *Join − accept* message [3].

Rejoin-request Message The end-device's *NetID* (the device's home network identification) and *DevEUI* are contained in the *Rejoin − request* type 0 or 2 message, which is followed by a 16-bit counter (*RJcount*0). With each Type 0 or Type 2 Rejoin frame sent, the counter *RJcount*0 is increased. Each time the end-device successfully processes a *Join − Accept*, *RJcount*0 is initialized to 0.

The Network Server must save the last *RJcount*0 value that each end-device used, which is known as RJcounto_last. If (Rjcounto <= RJcounto_last) *RJcount*0 SHALL never wrap around, then it disregards Rejoin-requests. The device will cease sending *ReJoin − request* type 0 or 2 frames if *RJcount*0 hits $2^{16} - 1$. The gadget might return to the Join state.

There is no encryption on the *Rejoin − request* communication. The duty-cycle of the device's Rejoin-Req type 0 or type 2 transmissions MUST always be less than 0.1%.

- **Rejoin-Request type 0** message is used to reconnect mobile device to a visited network in roaming situations. A static device's devAddr can also be changed or rekeyed using it. Static devices should send this message less frequently than mobile devices, which are anticipated to move between networks.
- **Rejoin-Request type 2** message is only meant to enable rekeying of an end-device. Only after receiving a ForceRejoinReq MAC instruction can this message be sent.

Fig. IV.12 – *Lora Re-join Request Type 0 or 2.*

- **Rejoin-Request type 1** The *Rejoin − Request* type 1 message includes the *JoinEUI* and the *DevEUI* of the End-Device, much like the *Join − Request*. Any Network Server that receives the *Rejoin − Request* type 1 message can therefore forward it to the End Device's Join Server. The *Rejoin − request* Type 1 can be used to reconnect to an End Device in the event that the Network Server loses all of its state. At least one *Rejoin − Request* type 1 message should be sent each month.

Fig. IV.13 – *Lora Re-join Request Type 1.*

Unlike the *RJCount*0 used for *Rejoin − request* type, the *RJcount*1 for *Rejoin − request* type 1 is a separate counter. Every time a *Rejoin − request* type 1 frame is sent, the counter *RJcount*1 is increased. The Join Server records the last *RJcount*1 value that each End Device has utilized, also known as RJcount1_last. *Rejoin − requests* is ignored if (Rjcount1 <= RJcount1_last).

For a given *JoinEUI*, *RJcount*1 will never wrap around. For a specific *JoinEUI* value, the transmission periodicity of *Rejoin − Request* type 1 must be such that this wraparound cannot occur during the Device's lifespan. There is no encryption in the Rejoin-request-type 1 message. The duty cycle of the device's Rejoin-Req type 1 transmissions must always be less than 0.01%. This message is only utilized when the server side context is completely lost.

IV.5.6.2 Rejoin-Request Message Processing

whenever a Re-Join-Request (type 0, 1, or 2) is processed by the network server and a Join-accept message is produced. Until it receives the first successful uplink frame utilizing the new security context, it must keep both the old one (keys and counters, if any) and the new one. Only then may the old context be safely removed.

In all situations, the network server's handling of the Rejoin-request message is comparable to that of a typical Join-request message. This is because the network server first decides whether to forward the message to a Join Server so that a Join accept message can be created in response.

Fig. IV.14 – *Lora Join-Accept Message Fields.*

If the network server wishes to change the device's network identification (roaming or re-keying), it will reply with a $join-accept$ message. Then, in the key derivation process, $RJcount$ (0 or 1) takes the place of $DevNonce$. Additionally, the network server may optionally include MAC directives in a standard downlink frame. This downlink will be transmitted on the same channel as the $Join-accept$ message it replaces, with the same data rate and delay [3].

IV.6 General LoRaWAN Security Features

LoRaWAN is designed with a strong focus on security, adopting sophisticated cryptographic mechanisms to protect the data transmission between end-devices and the network. At the core of its security architecture are two primary keys provided to each end-device: the network layer *keyNwkKey* and the application layer key *AppKey*, together termed as root keys [28]. These keys facilitate the usage of the Advanced Encryption Standard (AES) algorithm for robust data encryption and authentication at both the network and application layers.

Secure communication is further reinforced by generating two session-specific keys from the root keys: the Application Session Key *AppSKey* for encrypting and decrypting the payloads, and the Network Session Key *NwkSKey* for protecting communication between endpoints and the network server.

The security protocol in LoRaWAN operates on two principal layers. Payload encryption is conducted using AES Counter Mode (*AES-CTR*) with the 128-bit *AppSKey*, while the integrity of the messages is upheld through a Message Integrity Code *MIC*, computed using AES Cipher-based Message Authentication Code (*AES-CMAC*) with the 128-bit *NwkSKey*. A frame counter provides defense against replay attacks, and the *MIC* guards against tampering. This dual-layer protection ensures the confidentiality, integrity of data within the network, and the authenticity of message received network and application servers from end-devices.

IV.6.1 Confidentiality of Messages

ED sends the join-request message without encryption. Therefore, this communication can be read by any device that has the ability to intercept it. In the join-accept message, the fields encrypted by NS are $AppNonce, NetID, DevAddr, DLSettings, RxDelay, CFList$, and MIC (details of calculating the MIC field are presented in Section IV.6.2, page 110). The $AppKey$ with aes128_decryp is used for encryption as follows:

AES128_Decrypt(AppKey,AppNonce | NetID | DevAddr | DLSettings | RxDelay | CFList | MIC).

Since the NS encrypts the message using the aes128_decrypt function, the ED must decrypt it using aes128_encrypt. In this manner, ED only needs to implement the aes128_encrypt function for the entire encryption procedure.

Except for *FRMPayload*, every field in a dataUp/dataDown message is transmitted in clear text in LoRaWAN v1.0.x. Actually, based on the value of the *FPort* field (0 for MAC instructions or [1...2,3] for application data), this field comprises the data application sent by AS or MAC commands sent by NS. The *AppSKey* is utilized for encryption unless *FRMPayload* contains MAC commands, in which case the *NwkSkey* is used.

As seen in Fig.IV.15, the *FRMPayload* field is encrypted using *AES* in a counter (*CTR*) mode. In the *CTR* mode, a stream of keys is generated and then XORed with *FRMPayload*.

The following steps describe how encryption is done:

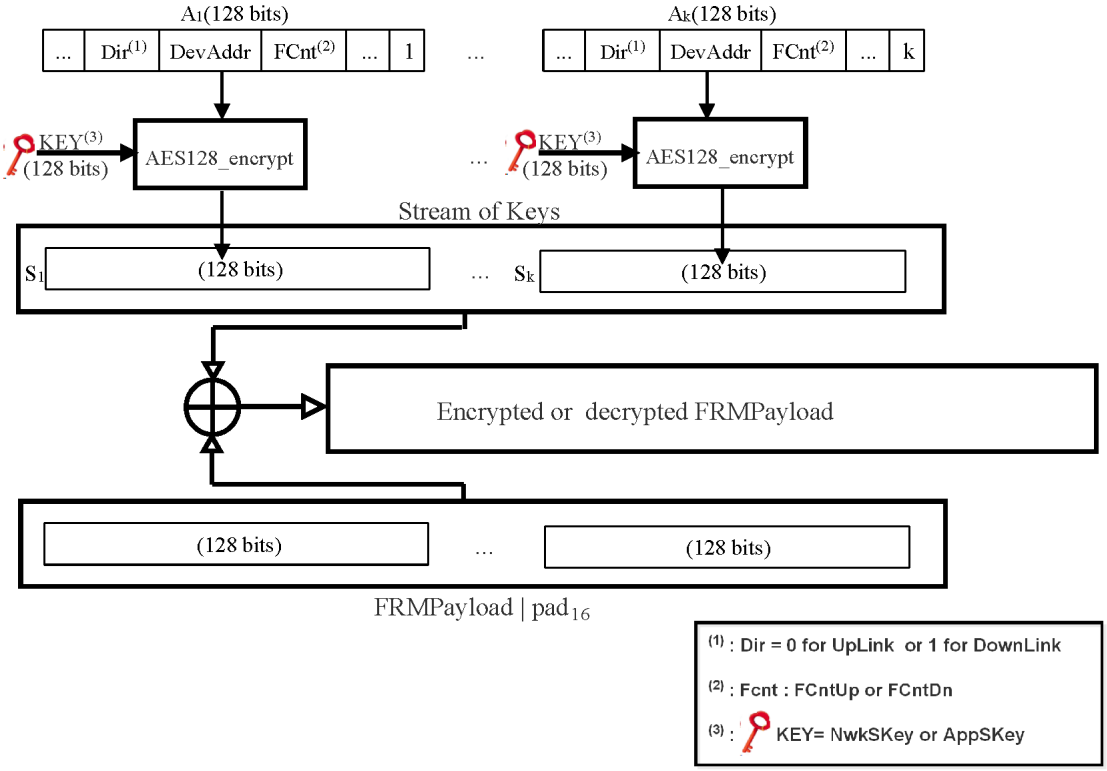


Fig. IV.15 – *AES 128 in CTR mode.*

• Defining a sequence of 128-bit blocks A_i for each message, with i = 1, ..., k indexing the blocks and k representing the total number of blocks needed, which is the ceiling of the payload length divided by 16 :

$$k = ceil[length(FRMPayload)/16].$$

Each A_i block is composed of various elements including the direction of the data *Dir* (with 0 indicating uplink and 1 indicating downlink), the device address *DevAddr*, and the corresponding counter $ctr(FCntUp$ for uplink or $FCntDown$ for downlink), along with the block index i.

These blocks are then individually encrypted to form a corresponding encrypted blocks S_i, i = 1, ..., k.

• Calculating a stream of keys (a sequence S of blocks S_i) by encryption the sequence of blocks A_i , using the AES algorithm 3.

S = S1 | Si | ... | Sk
S_i = AES128_Encrypt(K, A_i) i = 1 k
K = *NwkSKey* or *AppSkey*

• The *FRMPayload* is then encrypted, to produce *EncFRMPayload*, by xoring *S* with the *FRMPayload* as presented in Algorithm 3.

Algorithm 3 AES128-CTR for encrypting LoRaWAN frame payload using AppSKey.

Input: FRMPayload, AppSKey, keySize
Output: EncFRMPayload
Initialisation: k = ceil[length(FRMPayload)/16]
keySize = 128
function AES_CTR(FRMPayload, AppSKey, keySize, k):
  for $i = 1$ to k **do**
    A_i = oxo1||oxoo||oxoo||oxoo||oxoo||Dir||DevAddr||ctr||oxoo||i
    $S_i = AES_{CTR}$(AppSKey, Ai)
  end for
  S = $[S_1][S_2]...[S_k]$
  EncFRMPayload = S $\oplus$FRMPayload
  Return *EncFRMPayload*
end function

IV.6.2 Calculating the Message Integrity Code (MIC)

Two session keys is used to secure all LoRaWAN traffic . With $AES - CTR$, every payload is encrypted using the 128-bit Application Session Key $AppSKey$. To prevent packet replay, each frame has a frame counter. Additionally, each frame contains a 32-bit Message Integrity Code (*MIC*) to prevent packet manipulation. *MIC* is computed with *AES-CMAC* (AES Cipher-based Message Authentication Code), using the 128 bits network session key $NwkSKey$.

However, Fig.IV.16. illustrate the signing / encryption process that the MAC header, Frame header and encrypted payload are protected by a Message Integrity Code (*MIC*) derived from them and *AES*, where The computed *MIC* is then added to the end of message itself for message integrity checking. Once the message content is falsified, or delivers by a fake device, the *MIC* calculated by the network server itself will not be equal to the receiving *MIC*. In addition, the data security between an end-device and the application server is insured by the fact that the end-device encrypts plaintext by utilizing 128-bit AES algorithm with $AppSKey$, and generates a message with the ciphertext as the payload.

The application server decrypts the ciphertext with the same $AppSKey$. It is worth mentioning that each pair of end device and network server (application server) has an unique $NwkSKey$ ($AppSKey$).

Following the encryption of the frame payload, an integrity check is performed by calculating the 4-bytes *MIC* for messages transmitted within the *LoRaWAN* network using the *AES128 - CMAC* mode as presented in Algorithm 4.

Algorithm 4 AES128-CMAC to generate MIC of LoRaWAN frame payload using NwkSKey.

Input: EncFRMPayload, LoRaWAN_Header, NwkSKey, keySize
Output: MIC
Initialisation: k = 128
keySize = 128
function AES_CMAC(EncFRMPayload, NwkSKey, keySize):
  MIC = AESCMAC (NwkSKey, LoRaWAN_Header||EncFRMPayload)
    Return *MIC*
end function

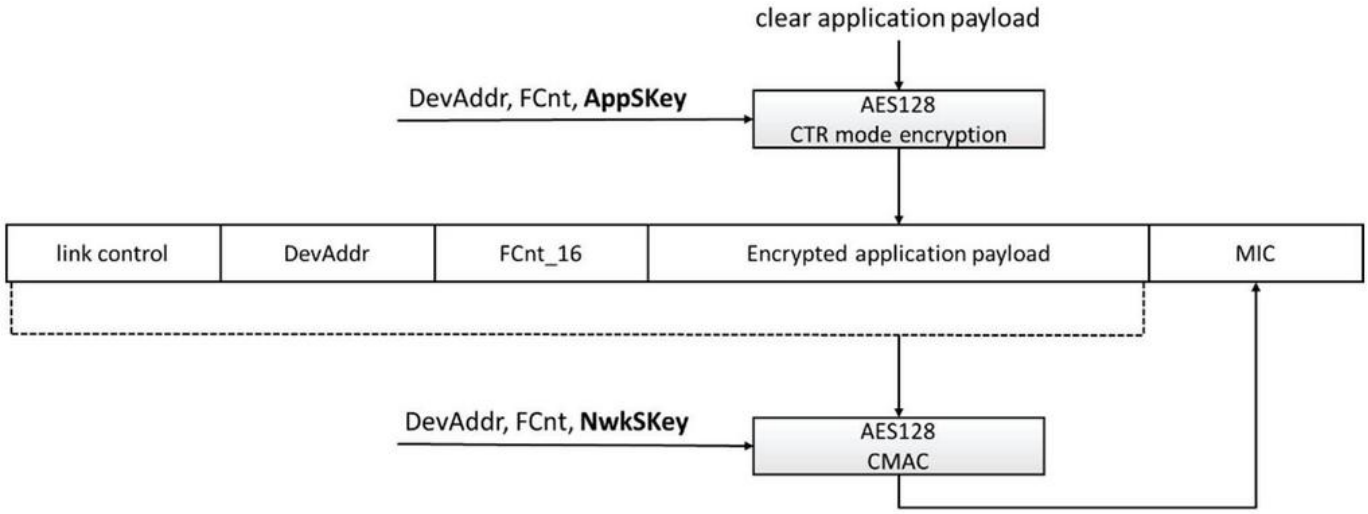


Fig. IV.16 – *Frame authentication and encryption.*

After calculating the MIC, the LoRaWAN frame is constructed according to Equation IV.12 as follows [106]:

$$LoRaWANFrame = LoRaWAN\_Header||EncFRMPayload||MIC. \quad \text{(IV.12)}$$

The integrity and validity of a message are guaranteed by the Message Integrity Code (MIC). After the payload for each packet (entire PHYPayload) has been encrypted, the message integrity code is determined. The MIC for each message type in LoRaWAN 1.0.x and 1.1 is determined by the fields listed in Table IV.5.

Table IV.5 – *Fields used and the key to calculate the MIC for each message type in LoRaWAN 1.0.x and 1.1.*

LoRaWAN version	Message Type	Key	Fields
1.0.x	Join-request	AppKey	MHDR | AppEUI | DevEUI | DevNonce
1.0.x	Join-accept	AppKey	MHDR | AppNonce | NetID | DevAddr | DLSettings | RxDelay | CFList
1.0.x	Data messages (up and down)	NwkSKey	MHDR | FHDR | FPort | FRMPayload
1.1	Join-request	NwkKey	MHDR | JoinEUI | DevEUI | DevNonce
1.1	Join-accept	JSIntKey	MHDR | JoinNonce | NetID | DevAddr | DLSettings | RxDelay | CFList
1.1	Rejoin-request Type 0 and 2	SNwkSIntKey	MHDR | Rejoin Type | NetID | DevEUI | RJcounto
1.1	Rejoin-request Type 1	JSIntKey	MHDR | Rejoin Type | JoinEUI | DevEUI | RJcount1
1.1	Data messages (uplink)	FNwkSIntKey & SNwkSIntKey	MHDR | FHDR | FPort | FRMPayload
1.1	Data messages (downlink)	SNwkSIntKey	MHDR | FHDR | FPort | FRMPayload

When a LoRaWAN 1.1 device is provisioned with a LoRaWAN 1.0.x Network Server, the MIC of the uplink and downlink data messages are computed with the $FNwkSIntKey$.

In summary, LoRaWAN's employment of *AES - 128* with *CMAC* and *CTR* modes establishes a robust, secure communication framework that strikes a balance between security imperatives and the operational efficiency of network connected devices. However, it is important to note that this system is still susceptible to various types of attacks, necessitating ongoing vigilance and enhancements in security measures to safe guard against potential vulnerabilities.

IV.7 Related Works

The cryptographic security in LoRaWAN is good as it uses *AES-128*. Is considered as being the block cipher of choice for future applications. However, that does not mean that the communication protocol is secure. Butun et al. [107] conclude that there are multiple attack vectors to LoRaWAN and that the security is dependent on the implementation. The authors state that there are a few critical mechanisms in the implementation that needs to be considered.

There has been ample work on LoRaWAN. The literature shows that LoRaWAN version 1.0 has some security vulnerabilities. Many of these vulnerabilities have since been fixed in version 1.1 and has improved the security of LoRaWAN.

Based on physical layer security in LoRaWAN, the authors [108], suggested using PHYSEC-based key management. When compared to other key management techniques, the authors' research revealed that it can be a good alternative to the key management solutions that are already in use while using less energy.

The work presented in [109] elaborated an experimental performance analysis of the OTAA procedure using a real LoRaWAN deployment in the field, with the objective of analyzing the delay in activation and energy consumption on a large-scale LoRaWAN. The authors came to the conclusion that high network traffic is a big problem in *OTAA* activation. Long activation delays occurred (50% of the devices took more than 2 hours to activate). There was also a high number of packet retransmissions. Three main factors affect the performance of the *OTAA* procedure: (i) collisions; (ii) retransmissions; and, (iii) communication request work cycle.

The communication layer between stack servers is the focus of another proposed secure LoRaWAN backend [110]. Ellipitic Curve Cryptography (ECC) is the basis for their proposal, Server Session Key Generation (S2KG), which uses it to generate network session keys.

There have also been proposals for a LoRaWAN security evaluation testbed. To verify a unique Adaptive Data Rate (ADR) spoofing attack and the vulnerability of missing beacon authentication in Class B operation, for example, ChirpOTLE [111] is utilized. By updating the LoRaWAN definition, it enabled the authors to provide defenses against both attacks. A different proposal [112] integrates software-defined radio (SDR) and GNU Radio with standalone LoRaWAN transceivers. It enables them to replicate an attack from the middle.

For LoRaWAN, Tsai et al. suggest a hardware low-power AES data encryption architecture (LPADA) [113]. The main idea behind the concept is to employ a low-power lookup table to replace AES and to better control the power distribution of idle AES logic over its several rounds.

The cost of employing various *AES* key sizes with varying payload sizes on a commercially available LoRaWAN device is examined in [114]. The findings demonstrate that employing longer key sizes is a workable way to improve LoRaWAN security and that costs in terms of delay and energy usage are moderate.

Naoui et al. [115] analyzed the security of the LoRaWAN 1.0 protocol. The authors claimed that the LoRaWAN protocol is easily attacked by two possible attacks. The first one is the parameter DevNonce which is a 16-bit counter starting at 0 when the end-device is initially powered up and which is incremented by one with every join-request. Since the DevNonce is not encrypted in the join-request message, the attacker can utilize previous join-request messages to launch replay attacks. The second one is the parameter AppNonce which is generated when the network-server receives the join-request message from the end-device. The AppNonce is then passed to the end-device and application-server for the purpose of mutual authentication. An attacker can initiate the corresponding join acceptance message and send it to the end-device in the subsequent message. In [115], the authors designed a trusted third-party computer, which is utilized to dispatch the session key for network-server and application-server. The trusted third-party computer creates a timeline, and network-server stores the timeline when it receives a join-request message so as to prevent a replay attack.

A proposal was made by Jakub et al. [116] to incorporate the fog computing concept into

LoRaWAN. In order to achieve greater efficiency for massive volumes of data, the core concept of this paradigm is to deliver data processing and storage closer to the end devices. The authors suggested three fog computing-based IoT network designs in this regard. To determine which of the suggested architectures was best, each architecture was simulated and compared in terms of service time. By cutting down on latency, bandwidth, and efficiency, fog computing offers numerous advantages to IoT domains. Security concerns, however, must not be disregarded.

An analysis of security attacks on battery energy efficiency is presented in article [117]. By flooding the network with additional devices, the authors were able to disrupt packets being sent to Gateway (GW) and execute a Denial of Service (DoS) end node attack. Using SF 7 and SF 12, the authors analyzed battery usage without any assaults; SF12 had an 18-fold increase in consumption over SF7. A denial-of-service attack was used to repeat the measurements. The data collected demonstrated that the battery usage in this instance increased up to five times, significantly reducing the battery life.

IV.8 Implementation of AES-128 algorithm under NS3

Despite the enormous research and the various works carried out by researchers and labs, in all the literatures according to our knowledge, we do not find the deployment of cryptography under the $NS-3$ simulator. So our goal is to implement the AES algorithm under the $NS-3$ simulator, and therefore let's examine the performances of the Lora network such as PDR, ToA, Energy and CPU Time [118], for this we will work as follows:

As illustrated in Fig.IV.17, at the physical layer, the packet will be split and take just the payload in plaintext, then encrypt only the payload according to the specification [3], using the AES-128 bits encryption algorithm based on the library accessible via the following link [119], once the ciphertext is successfully received by the receiver (by the network server), it will be decrypted with the same encryption key (in symmetric encryption as AES standard, the encryption key is the same decryption key), and finally the decrypted uplink message will be converted, and displayed in plaintext.

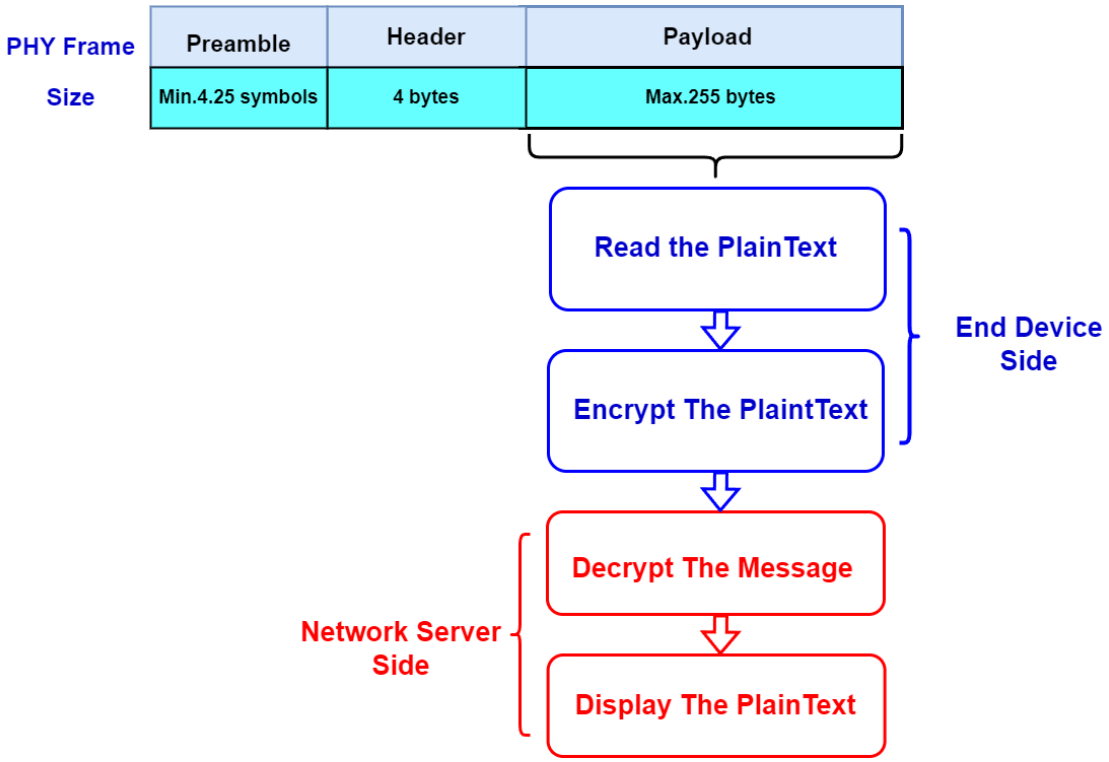


Fig. IV.17 – *AES Flow Implementation.*

Under NS-3, there are two sending modes, either one shot, i.e., during the entire simulation the transmitter sends just one packet, while the second mode is periodic sender, the transmitter sends several packets in a period of time during the duration of the simulation (the 2nd case is the choice as the results obtained in the two previous chapters, our script sends a packet of 12 bytes during simulation time equal to one hour).

Fig.IV.18 summarizes the existing class diagram of the NS-3 simulator, and highlight the integration of AES encryption algorithm in this simulator. The implementation of AES standard in NS-3 is done by adding new functions and modifying the existing NS-3 classes and functions. The MAC layer, which is responsible for transmitting the packets, and the helpers are charged with initializing the configuration parameters of each scenario, including the kind of encryption (AES-128, AES-192, and AES-256). The encryption keys are configured in the *PeriodicSenderHelper*.

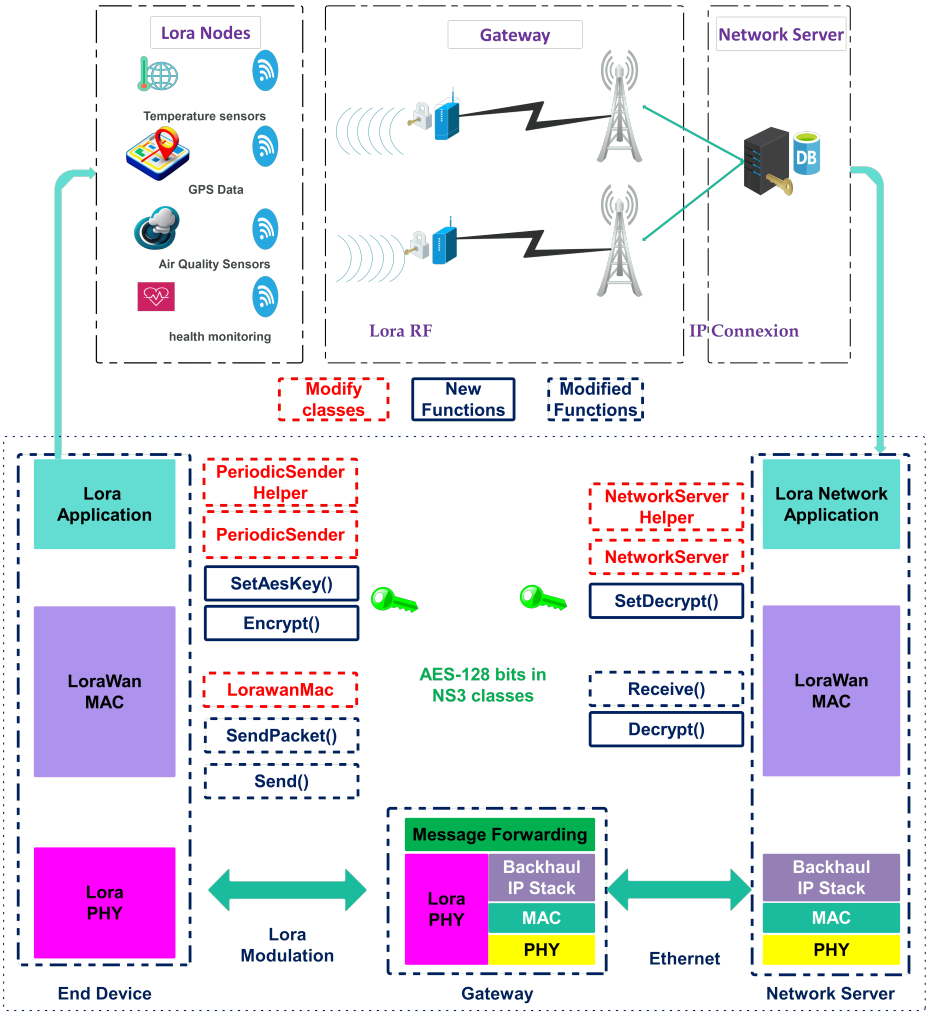


Fig. IV.18 – *AES-128 integration in NS-3 structure*

This operation calls the *setAesKey*() function and instantiates an object of the *PeriodicSender* class, which is responsible for encryption. Through the *encrypt*() function, the encryption is done before the packet will be sent. The PeriodicSender class even implements the *SendPacket*() function to call an object of the *LorawanMac* class, which implements another *Send*() function whose role is to send messages.

On the other side, on the Network Server side, the same steps are done for the decryption once the message is received. The NetworkServerHelper class initializes the shared parameters (symmetric encryption) by the *SetDecrypt*() function and calls an object of the NetworkServer model class to do the necessary.

- Receive the message by the *Receive*() fonction.
- Remove the headers.
- Decrypt the message via the *Decrypt*() fonction.

IV.8.1 Results and Discussions

The result of the execution this scenario is displayed on the terminal illustrated in Fig.IV.19 highlighted in three (03) steps:

- Number 1: reads the plaintext and displays it in hexadecimal.
- Number 2: encryption processing with the AES-128 bits algorithm, then sending a message and displaying ciphertext.
- Number 3: finally decrypts the received ciphertext and displays it in plain text.

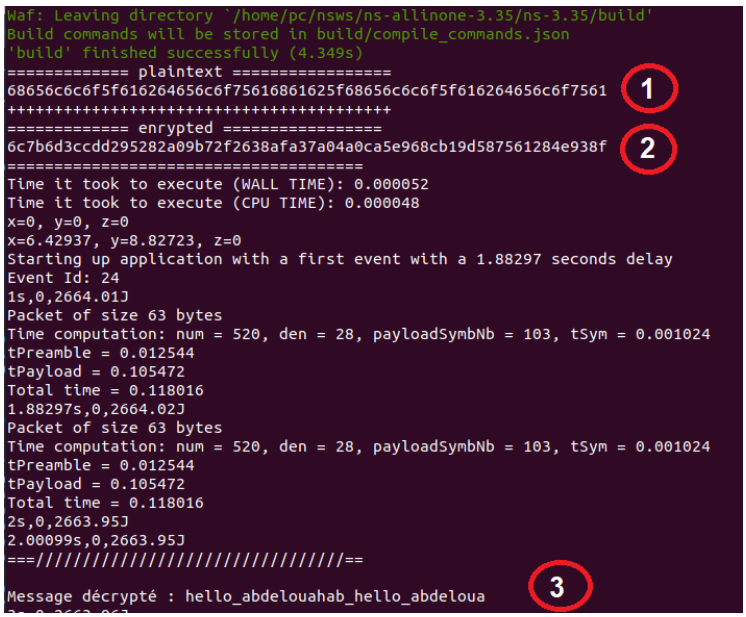


Fig. IV.19 – *The execution result on the terminal.*

To measure the energy consumption, Time on Air, and Packet Delivery Rate induced by the cryptographic primitives used in the LoRaWAN stack according to various packet sizes, based on the parameters used in Table.IV.6, successfully gathering data from multiple simulation scenarios.

Table IV.6 – *Simulation Parameters.*

Parameter	Value	Unit
N of Nodes	50	-
Radius	20	Meter
SF	7	-
Bandwidth	125	kHz
Packet Size	12, 24, 32, 64, 128, 192, 216	Bytes
Simulation Time	3600	Second
Energy Initial	200	mAh
Batterie PD2032	2664	Joules
	3.7	Volts
Simulator	NS-3 (Version 3.35)	-
Operating System	Ubuntu 24.4 64 bit	-
Machine	Intel Xeon(R) CPU E5-2620V4	-
	32 Go RAM	

This information is used to analyze the performance metrics, and also it provides efficiency for the cryptographic method. Consider a scenario that sends a packet of 12 bytes with SF7 and bandwidth equal to 125 kHz, the maximum transmission power (+14 dBm), for a simulation time of 3600 seconds, the PD2032 battery model has a initial energy of 200 mAh and 2664 joules, with the end devices randomly distributed in a radius of 20 meters around one gateway (GW).

In order to demonstrate the impact of AES-128 cryptography, the difference with and without AES-128 bits in value is shown in Table.IV.7.

Table IV.7 – *Time difference in ToA with and without AES-128 bits.*

	12 Bytes	24 Bytes	32 Bytes	64 Bytes	128 Bytes	192 Bytes	216 Bytes
ToA Without AES-128	0.056476	0.071836	0.087196	0.133376	0.225536	0.327936	0.358656
ToA With AES-128	0.0567319	0.0721809	0.0876003	0.133918	0.22656	0.329472	0.360356
Difference (Sec)	0.0002559	0.0003449	0.0004043	0.000542	0.001024	0.001536	0.0017

Fig.IV.20 shows the Time on Air (ToA) for different payload lengths (in bytes), using a radius equal to 20 meters and a single GW. In LoRaWAN, time on air defines the elapsed time on air for a LoRaWAN packet between the end device and gateway. Time on air for different configurations for each packet can be calculated using a formula provided in LoRaWAN specifications [26]. As expected, payload length plays an important role on the ToA.

The Time on Air increases with increasing packet size with and without AES-128 encryption; there is a slight difference between the two histograms.

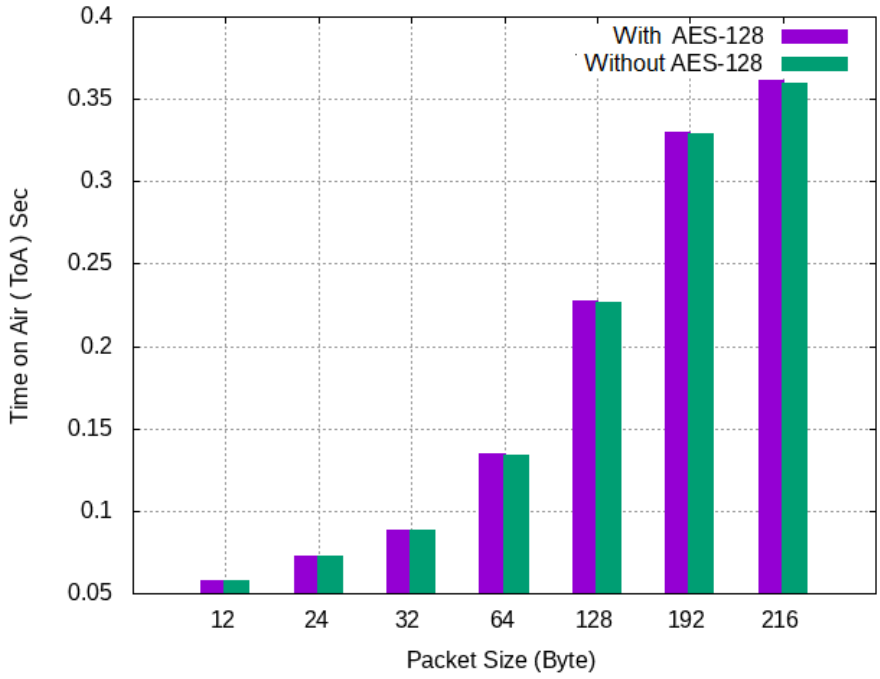


Fig. IV.20 – *Time on Air (Sec) vs. Packet Size*

Since Time on Air is directly related to the amount of energy a node needs to spend to transmit the data packet, it is important to determine the battery life of a node. Simulate

different scenarios by increasing the packet size $\in \{12, 32, 64, 128, 192, 216\}$. Fig.IV.21 examines the energy remaining of nodes in a network during one hour of simulation, focusing on packet sizes, with energy measured in joules. A significant decrease in the remaining energy occurs when the packet size increases from 64 to 216 bytes. Highlighting the impact of encryption AES-128 bits, packet size, and communication frequency on energy usage.

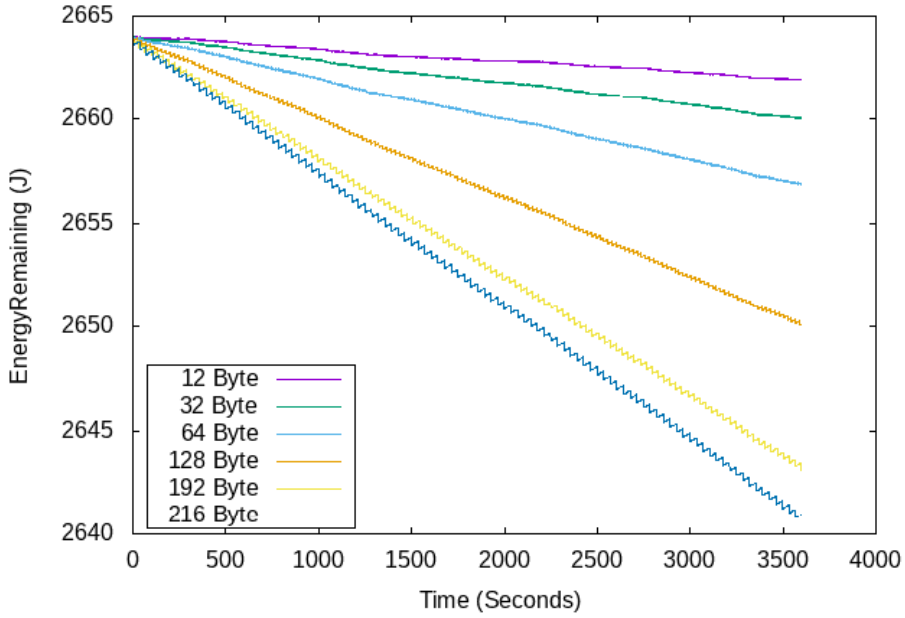


Fig. IV.21 – *Energy remaining vs. Time (Sec)*

Packet Delivery Ratio (PDR) is the sum of received packets divided by the total packets sent. In Fig.IV.22, the PDR is calculated based on the packet size. Plotted seven different packet size values with and without AES-128 bits. The Table IV.8 shows the difference in value.

Table IV.8 – *Packet Delivery Ratio with and without AES-128 bits.*

	12 Bytes	24 Bytes	32 Bytes	64 Bytes	128 Bytes	192 Bytes	216 Bytes
Packet sent with AES-128	30873	25300	20769	13600	8050	5550	5050
Packet received with AES-128	25697	19027	14724	8154	3166	1665	1395
Packet sent without AES-128	30862	25180	20840	13650	8010	5750	5150
Packet received without AES-128	25697	19027	14792	8192	3182	1782	1442
PDR with AES-128	83.23 %	75.50 %	70.89 %	59.95 %	39.32 %	30 %	27.62 %
PDR without AES-128	83.26 %	75.56 %	70.97 %	60.01 %	39.72 %	30.99 %	28 %

As expected, as the packet size increases, the PDR also decreases for both hystograms, either with or without AES-128 bits encryption. Note that for the smallest size of 12 bytes with the lowest airtime, almost 83% PDR is achieved. While for the bigger size of 216 bytes, the PDR value drops to only less than 28% due to the longer packet transmission time, packets are more vulnerable to collisions. In addition, it should be noted that the transmission of packets encrypted with AES-128 bits impacts the transmission time, which is longer than without using AES-128 bits cryptography, and which increases the propensity to collisions and therefore directly impacts the PDR. This confirms the results obtained in Fig.IV.20 when the packet size increases.

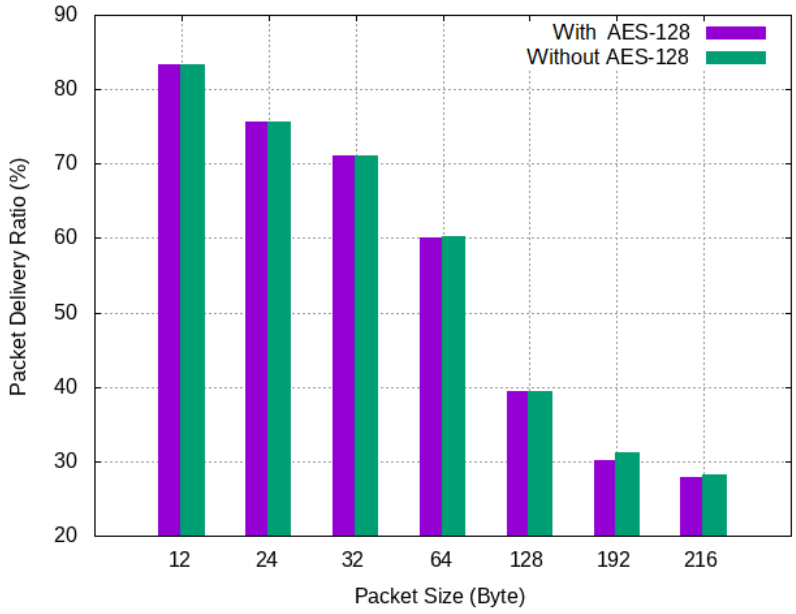


Fig. IV.22 – *Packet Delivery Ratio vs. Packet Size*

Fig.IV.23 shows the CPU time consumed during encryption in seconds compared with the packet size using the AES-128 bits standard. There is a linear increase up to the maximum size, which took 24 seconds for a packet of 216 bytes.

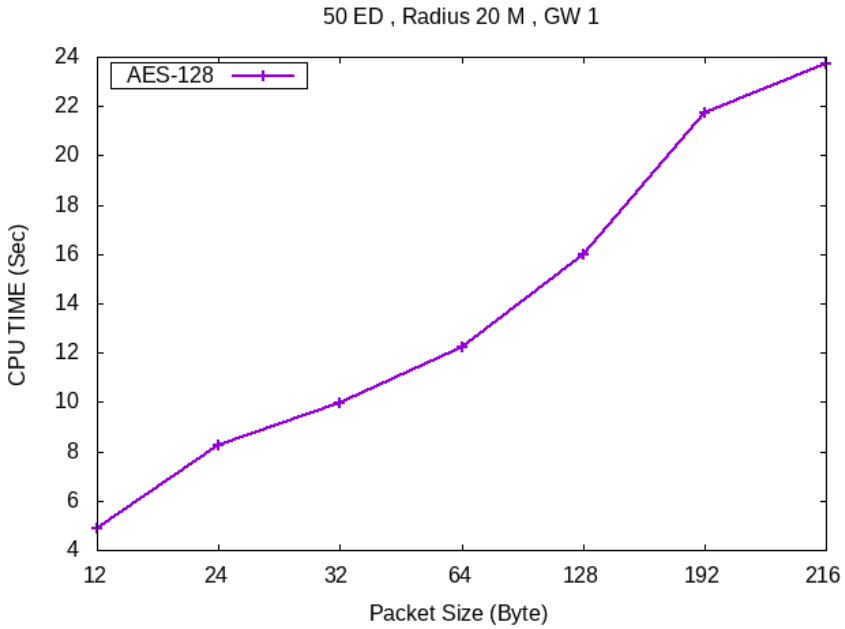


Fig. IV.23 – *CPU Time vs. Packet size with AES-128*

IV.9 Vulnerabilities and Attacks

LoRaWAN specifies a number of identifiers for devices. All end-devices have a 64-bit unique identifier called Device Identifier (*DevEUI*) which is set by vendors or developers. In addition, all communication is done using 32 bit device address. Another identifier, called Application Identifier (*AppEUI*), uniquely identifies the application provider of the end-device.

In order to provide many levels of encryption in LoRaWAN, the cryptographic security is

handled by $AES-128$ operating in CTR mode. Network and application level packet security is achieved by LoRaWAN through the use of distinct device, network, and application keys. This maintains the anonymity of application data while enabling intermediary nodes, such gateways and cloud routers, to carry out routing and network maintenance activities.

Two session keys, referred to as the Network Session Key ($NwkSKey$) and Application Session Keys ($AppSKey$), are generated using a 128-bit AES key known as the Application Key ($Appkey$). The message integrity code (MIC) is generated and verified by the network server using the shared NwkSkey between the end device. This generates a unique signature for each device and guarantees the integrity of the communications. Though it is used for encrypting and decrypting the application data payload, the $AppSKey$ and $NwkSkey$ are comparable. $NwkSKey$, $AppSKey$, and the messages' up-link or down-link counter are used by LoRaWAN to generate a key stream. In order to create the encrypted payload, each message is therefore encrypted using the XOR operation with the appropriate key from the key stream.

The security is guaranteed by the LoRaWAN protocol, which also offers encryption between the end device and the gateway. The payload length is constant, though, both before and after encryption. A malevolent actor might exploit this in conjunction with overflowing counters to recover the key stream from the encrypted communications.

IV.9.1 LoRaWAN Possible Attacks

The confidentiality, integrity, and availability of data on LoRaWAN might be compromised by a number of possible attacks on security. To survive such attackers in the network, LoRaWAN critical security models must be updated on a regular basis. Some of the current attacks are covered in this subsection:

IV.9.1.1 Authentication attacks

In this section, we present the LoRaWAN authentication attacks, along with proposed countermeasures.

❖ **Man in the middle attack (MITM)** The scenario of MITM attack between the ED and the GW is described below and illustrated in Fig.IV.24:

1. The device sends a message.
2. The attacker intercepts it.
3. The attacker changes the payload.
4. Using the compromised NwkSKey, the attacker signs the message with a valid MIC.
5. The attacker sends the modified message to the GW.
6. The NS checks the message's MIC with the NwkSKey and validates it before forwarding it to the IoT platform.
7. The AS receives the modified message.

Since the attacker knows of the $AppKey$, they will collect the join-procedure messages (join-request and join-accept) after compromising the $AppSKey$. They will then calculate the $NwkSKey$, which can be found using equation IV.4.

The MITM attack is executed by the attacker after he obtains the $NwkSKey$. He intercepts the messages that are communicated between the ED and the NS, which allowed him to change the communicated messages or inject new ones—in fact, this is done prior to the integrity checking

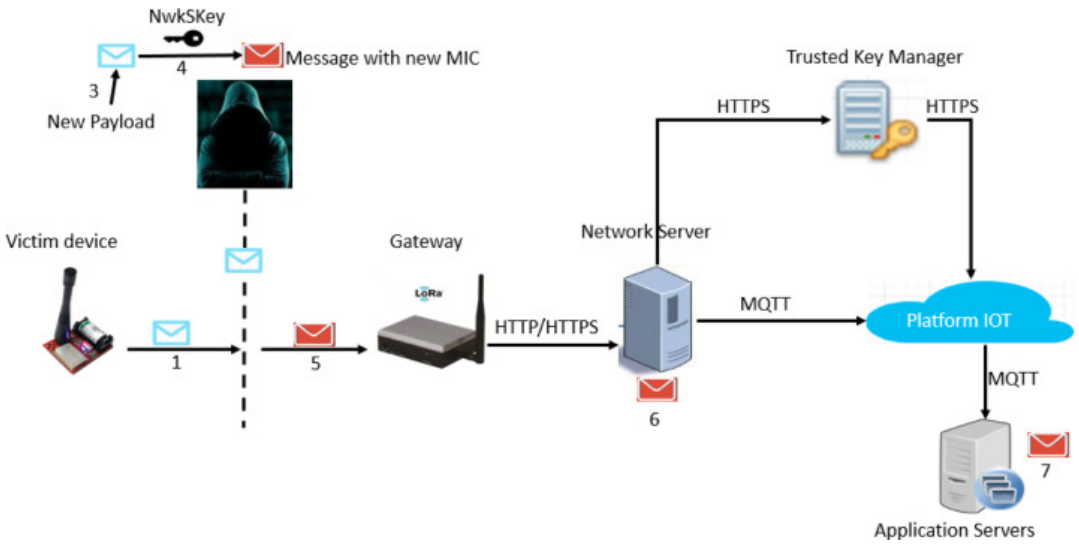


Fig. IV.24 – *Lora Join-Accept Message Fields [120].*

phase. The $NwkSKey$ is used to generate a new MIC for the modified and new messages, so these attacks can be carried out between the GW and the NS or between the ED and the GW.

The MIC is related to the obtained $h(AppKey)$ in a non-linear and non-invertible connection to prevent this attack. In the suggested solution [120], *h(MIC XOR h(AppKey))* is sent in place of $h(AppKey)$. The MIC is connected to $h(AppKey)$ in a non-invertible way, which prevents the unauthorized GW from altering and disguising the legitimate device. In fact, each alteration will produce a distinct MIC that is inseparable from $h(AppKey)$. It will thus be extremely challenging to carry out this attack. To verify the ED request, the NS makes use of the MIC from the $join-request$ message and the stored $h(AppKey)$.

IV.9.1.2 Confidentiality Attacks

Confidentiality refers to securing data from unauthorized sources. To put it another way, if the data is seen by those who are supposed to have access to it, it could no longer be secret and could instead become known to an enemy. This section outlines two important aspects of confidentiality in the context of LPWAN, namely data confidentiality and privacy.

❖ **Eavesdropping Attack** This attack includes an intruder listening passively over a weak link in the network to learn about the transmission of data. Cloning devices or using keys to sniff network traffic might be a vulnerability. Intruders can read and manipulate the packets being sent and received if they get their hands on the key. The gateways are responsible for gathering packets from end devices and forwarding them to the network servers within the context of LoRaWAN threats. Consequently, a rogue gateway in LoRaWAN has the potential to cause security breaches in the network.

❖ **Privacy** In the context of the Internet of Things, privacy is a crucial component of data protection as it prevents unauthorized parties from accessing personally identifiable information (PII)—also known as personally identifiable information—that belongs to end users. Understanding and recognizing the correlation of devices at the network's edge is the focus of data privacy breaches. Attacks on privacy pose significant obstacles to the creation of innovative apps. Various types of algorithms [120] that solve privacy breaches are accessible in the literature.

IV.9.1.3 Integrity Attacks

Integrity is the basic component of data security as it guarantees to preserve the data accuracy, completeness and reliability in the network. It attempts to protect data during transmission from a source to a destination by preventing tampering, additions, or changes. Data integrity and system integrity are two more topics that are covered in the context of integrity.

❖ **Bit Flipping Attack** A low difficulty and highly motivated attack known as "bit flipping" involves the attacker analyzing the packet and then changing certain portions of the ciphertext without being able to understand it. The packet delivers improper information to the target point as a result of the ciphertext being altered. For LoRaWAN, end devices utilize counter mode to send the packet to the gateway. By using an XOR operation, the counter mode secures plaintext without changing the order, which makes it susceptible to bit flipping attacks.

❖ **Malware Attack** Malware attacks occur at the application layer and impact the system through a variety of risks, including worms, trojans, and ransomware. In this assault, the attacker monitors the user's activities and has the ability to contaminate data with viruses. In addition, if an attacker gains access to the application system, the end user may be removed.

IV.9.1.4 Availability attacks

This section presents and analyzes a collection of LoRaWAN availability attacks. Furthermore, a list of potential defenses against these attacks is suggested.

❖ **Sinkhole attacks** The primary objective of an illicit node in this kind of attack is to route network traffic to a certain node. Therefore, as seen in Fig.IV.25, the attacker will infect the network nodes to use a certain route that is promoted in the network. This assault impacts the system's availability and can be dangerous when coupled with other attacks. Such assaults may be identified and stopped with the use of an intrusion prevention and detection system (IDPS).

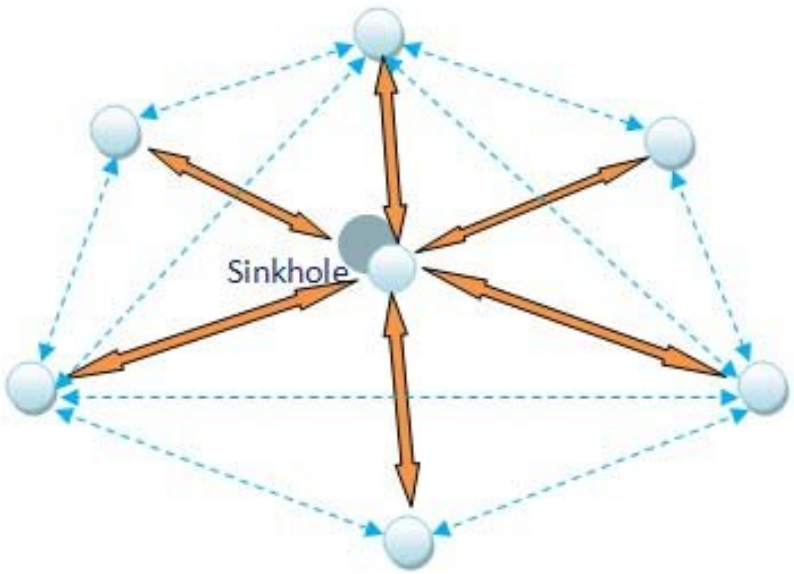


Fig. IV.25 – *Sinkhole attack topology.*

❖ **Down-link Routing Attack** By adding a compromised gateway to the network, this attack is carried out. When an attacker intercepts a successful transmission, it is known as a down-link routing attack. While the attacker eavesdrops and acknowledges the identical communication through the compromised gateway, the end device uses the authentic gateway to send an uplink message to the network server. As a result, receiving duplicate packets from both gateways may result in a downlink fault. Although the packets are de-duplicated by the network server, the established network between the gateway and the legitimate device might not be guaranteed.

❖ **Joint Accept Attack** LoRaWAN technology is severely impacted by this attack when packets are exchanged between the server and the node. While the gateway transparently forwards the packet from the end device to the network server, the end device uses the OTAA technique to connect with the network server. Upon receiving a request by the network server, it releases a join accept message through a gateway. The attacker with a motivation of this attack tries to reach the join accept message before the authentic end device and respond illegitimately to the network server. Consequently, the network for being exchanging the packet is established with unauthentic devices and fails the link between the node and the server.

❖ **Compromising Device and Network Keys** LoRaWAN uses network and application keys to offer end-to-end security. Nonetheless, the LoRa end-devices might be compromised by an attacker who has physical access. An attacker may take out the keys from a device if they succeed to have physical access to it. Usually, a LoRa radio module and a host microcontroller unit (MCU) make up end devices. The radio module uses an SPI or UART interface to connect to the host microcontroller. External hardware can be used to intercept data exchanges and commands between the radio module and the host [121]. The majority of radio modules available today do not have encryption functionality built in to protect interactions between the radio module and host microcontroller. In such situations, it is impossible to determine whether the host MCU or a malicious entity sent the orders that were transmitted to the radio module. Also a malicious entity could intercept all the data exchanges between the host MCU and the radio module and use this intercepted information to create a mock device with the same credentials or manipulate the data payload. Therefore, sensitive actions like defining security keys for each data transmission should not be carried out by application developers since this may expose important information to malevolent groups.

❖ **Beacon Synchronization Attack** Beacon synchronization is a predicament to Class B devices in LoRaWAN. In this attack, the attacker broadcasts fake beacons by means of setting up a rogue gateway in the network. In consequence, the Class B devices open multiple windows and receive down-link messages in a flooding fashion without synchronizing to the gateway. This kind of action tends the network to increase collision in packet exchanging. The authors in [122] propose that beacon synchronization may be resolved by employing a key at the gateway in order to solve this problem.

❖ **ACK Spoofing Attack** In LoRaWAN, a spoofing attack is a scenario in which an attacking node pretends as another to gain access to the network by falsifying data with an illegitimate approach. This attack is limited to only down-link and acknowledgment of the message. In this attack, the attacker first hands-on a down-link message and then manipulates it for any up-link confirm message flooding from the same device [123]. Spoofing attacks significantly affect the physical layer and have severe implications for network disruption and denial of service.

❖ **Replay Attack** A replay attack is an attack on security protocol, re-sending or repeating the valid data transmission by the malicious entity. This attack's primary objective is to trick the module or device by employing outdated network data or handshake messages. The entity

must be aware of the communication frequencies and channels in order to sniff data being transmitted between devices when conducting an attack on a wireless network.

In LoRaWAN, without *AppSKey*, it is impossible to decrypt transmissions between end devices and gateways since it encrypts the whole content of the message. Furthermore, it cannot be done without *NwkSKey* since altering the data will cause the *MIC* check to fail. Although the malicious entity can resend the message consecutively, using frame counters which are defined in LoRaWAN specifications these messages or attacks can be detected and discarded. When the end-device is turned on, both of these counts start at zero, and they go up with every message that comes in from the gateway or the device. A message is ignored if it receives with a frame counter that is less than the previous message as shown in Fig.IV.26. However, the LoRaWAN specification handling off frame counters is specifically left to the application and developer. Therefore, networks which do not track these frame counters could be vulnerable to replay attacks.

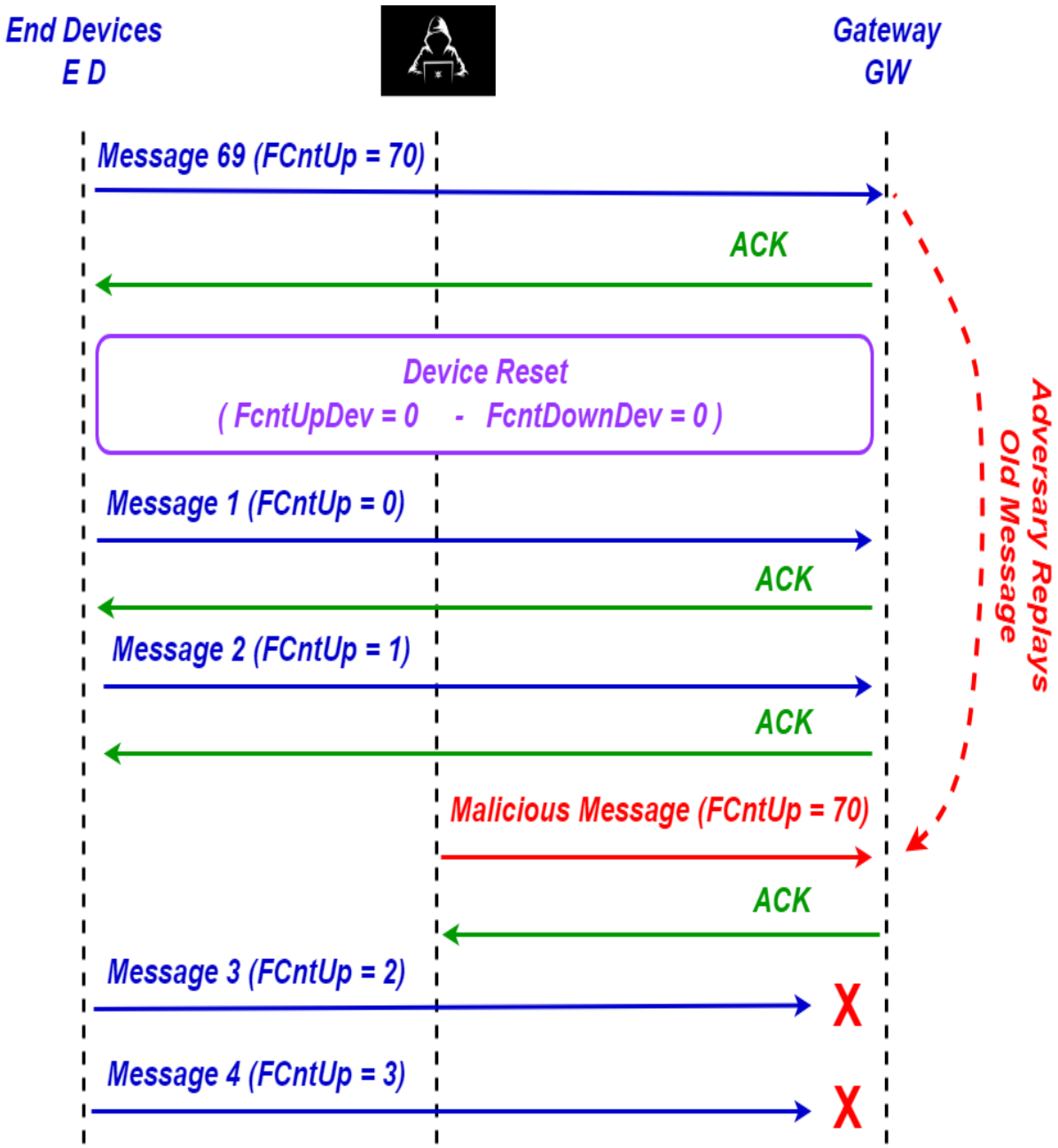


Fig. IV.26 – *Replay Attack.*

❖ **Wormhole Attacks:** This kind of attack involves a malicious device capturing packets from one device and sending them to another machine that is located far away in order to replay the captured packet [121]. Without having any prior knowledge of the network or cryptographic technique, a malevolent entity might easily initiate this. There are two types of devices that may be used to carry out a wormhole attack in a LoRaWAN network: sniffers and jammers. When the sniffer has successfully captured a packet, it alerts the jammer. No captured packet ever reaches the gateway, and the captured message's validation is still valid. It is possible to replay the recorded message at any time. It is sent to the application layer via the network server and gateway. As a result, ordinary communications that have been previously recorded but have never reached the gateway might be transmitted to it as though there were no alarm, and the essential alarm messages could be jammed. It is challenging to identify this assault in LoRaWAN networks because LoRaWAN messages lack time-related information.

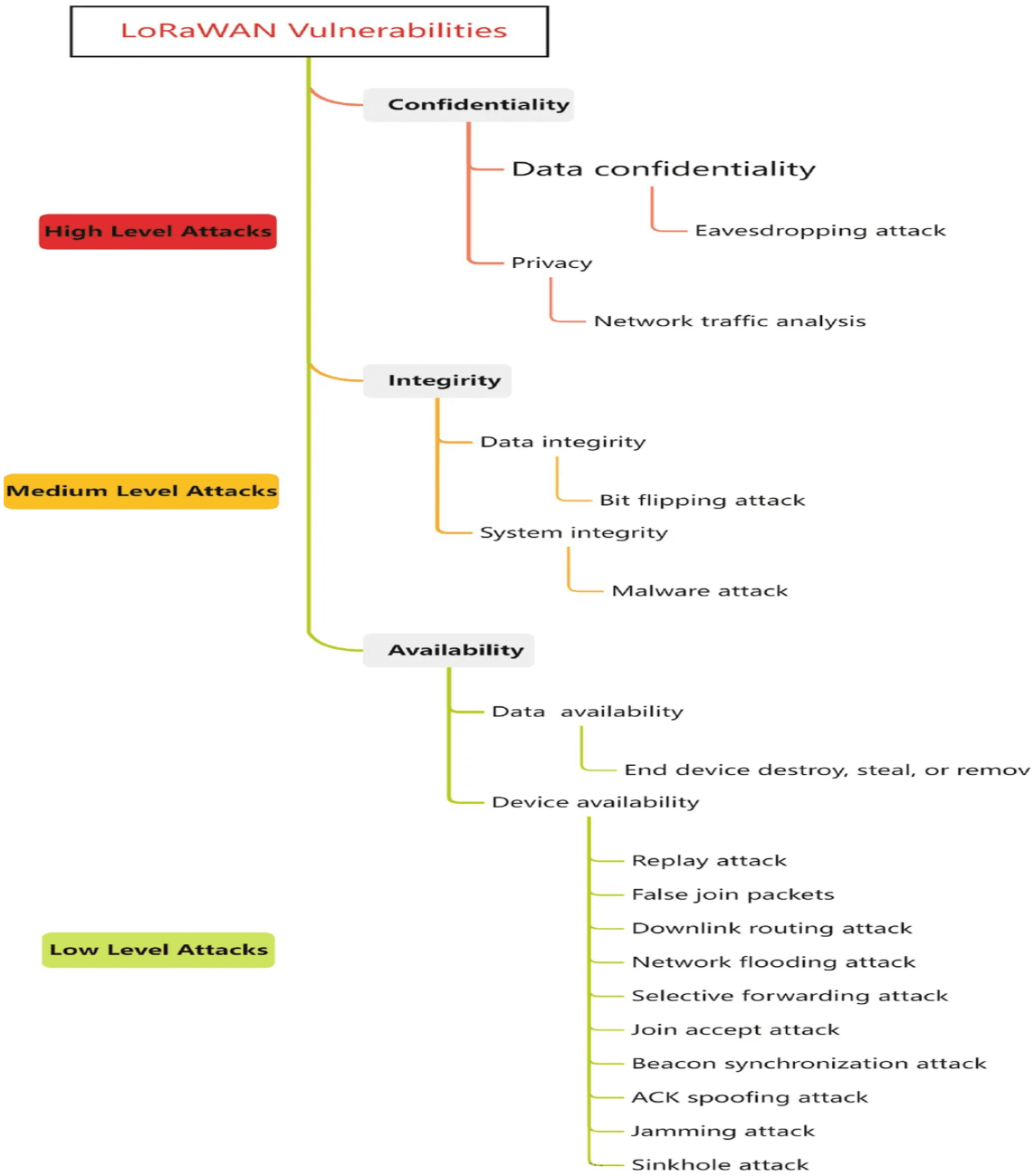


Fig. IV.27 – *LoRaWAN vulnerabilities [124].*

❖ **Jamming Attack** Jamming attack is the most critical issue in LoRaWAN technologies. In the jamming attack, the attacker first analyze transmitting frequency in the network and then tune the same frequency for the attacking node to disrupt communication, attackers have the ability to interfere with the radio transmissions. The disruption of communication could be by sending the number of bits, packets, and non-continues signal transmission to a particular channel. Jamming is easily performed in LoRaWAN as the end devices make use of CSS modulation technique advancement of the packet to the gateway. Detection of jamming attack in the network is quite challenging, however, by measuring the signal strength is made practical.

❖ **Rogue-Gateway Attacks:** A new class of attacks known as "Rogue-Gateway Attacks" was introduced by [124]. In these attacks, an attacker poses as a legitimate GW in the network and uses it to launch a variety of attacks on LoRaWAN networks, including as packet-dropping, black-hole, worm-hole, and selective forwarding.

IV.9.2 Countermeasures Addressing LoRaWAN Vulnerabilities

As the LoRaWAN system significantly affected from several security problems. Therefore, this sections lists suggestions, as shown in Fig. IV.28, that should taken into account while designing a secure LoRaWAN system. Suggestions to each security breach is discussed below:

- **Eavesdropping:** By exchanging one plain text for another, the attacker exploits this approach to directly compromise the network's confidentiality. For the purpose of defending the system from this attack, the packet should employ nonce.
- **Bit Flipping:** The message may be changed in this attack by utilizing the network layer. The application server may employ the message integrity code (MIC) to regulate this scenario.
- **Network Traffic Analysis:** Several programs and technologies that are necessary for effective communication are used to evaluate network traffic. If the packet-exchanging session is made changeable, this attack could become more complex.
- **Replay Attack:** The network's MAC layer suffers as a result of this assault. It takes place when the end device is being joined to the network server. Public key encryption, as described in the work [125], can stop the attack.
- **Sinkhole:** In a sinkhole attack, a single node is used to promote all traffic over a dedicated link. The intrusion detection and prevention system (IDPS) can be put into place within the network to take care of this problem.
- **Down-link Routing:** A compromised gateway can be used to carry out a down-link routing attack. Verifying the gateway's authenticity before to deploying it on the network will prevent this attack.
- **Jamming:** It is one of the most well-known types of attacks on RF-based communication networks. By analyzing the transmitted frequency and then altering the number of bits in the captured packet, a jamming attack may be carried out. Frequency changes might be useful in resolving jamming issues with LoRaWAN.
- **End Device Tampering:** A hard shell or tamper-resistant cover might make it more difficult to physically capture or tamper with end devices. Since the end devices are placed at the network's edge and are readily accessible to the attacker, this attack will probably occur.

- **Rogue-Gateway Attacks:** The authors of [124] introduced a unique method that enhances the LoRaWAN gateways' cybersecurity. Adoption of a public-key infrastructure (PKI) consisting of a two-tier certificate authority (CA) system is part of the plan. By utilizing the root-CA and intermediate-CA configuration, the 2-tier CA resolves the single-point failure scenario. The suggested approach effectively prevented harmful assaults, such as Selective Forwarding attacks, that emanated from the rogue-gateways, according to the authors' simulated results.

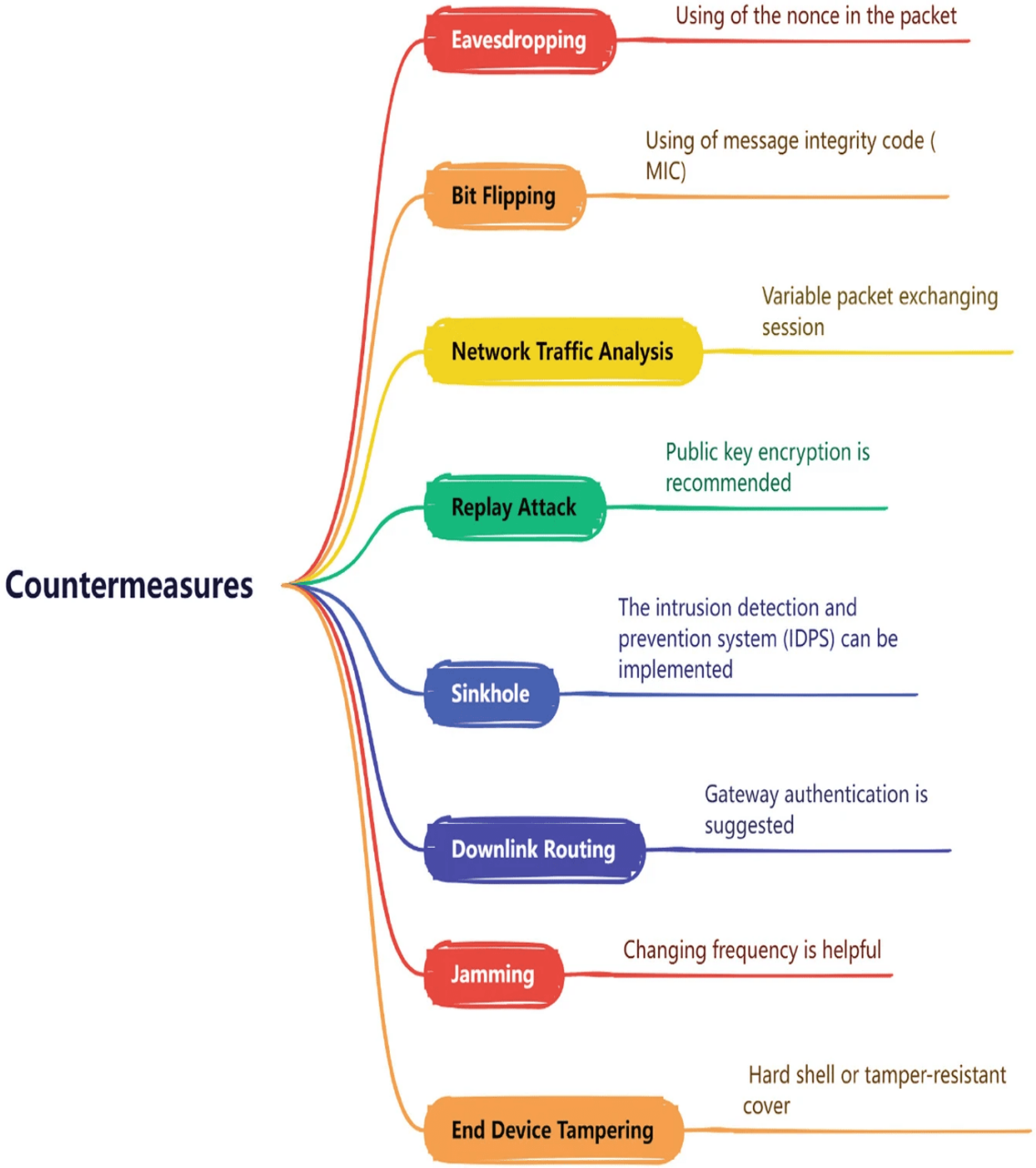


Fig. IV.28 – *Countermeasures associated with some of the LoRaWAN vulnerabilities [124].*

IV.10 Conclusion

In this chapter, we provided an analysis of the LoRaWAN network and its security. First, we presented the *AES* algorithm followed by a detailed structure of the lora packet, in section 4, we examined how LoRaWAN manages security by comparing the two methods (*OTAA* and *ABP*) of the LoRAWAN specification. We thus detailed the authentication and security rules of the two versions.

Next, in Section 8, we presented our contribution, a novelty in the entire literature: an implementation of the AES-128 bits encryption and decryption algorithm under the NS-3 simulator. We examined various results, such as PDR, ToA, Energy with various Packet Sizes, and CPU Time.

And finally, we provided an analysis of the vulnerabilities of the LoRaWAN network. It has been demonstrated in a number of studies that these networks have a number of vulnerabilities pertaining to the specification's definition and wireless nature. First, we observed that even with network-level authentication and end-to-end privacy, replay attacks and even privacy attacks are still possible. The lengthening of the ToA caused by LoRa modulation might therefore be used by attackers to carry out various jamming operations aimed at compromising the network's availability.

General Conclusion

The increasing deployment of IoT environments and the use of IoT applications in various fields are facing the key challenge of "security, improved quality of service (QoS)" in terms of reliability and energy efficiency. To respond to these challenges, the IoT relies on the use of a range of LPWAN (Low Power Wide Area Networks) technologies, notably due to their characteristics of long range, low throughput and low energy consumption. The aim of the work developed in this thesis is to address certain security and QoS management issues, by exploiting the performance of one of the LPWAN technologies, LoRa and its LoRaWAN MAC protocol. The objective is to propose solutions that both satisfy QoS requirements and optimize resources and energy consumption.

We began with a critical review of the main types of existing networks. This was followed by a discussion of the basic characteristics and operation of LPWAN standards, and the advantages and disadvantages of certain LPWAN standards. We then turned our attention to the LoRa standard, in particular its open MAC protocol (LoRaWAN). We identified a number of bottlenecks to be overcome for the LoRaWAN protocol concerning resource allocation and channel access.

After reviewing the state of the art of the various proposals published in the literature relating to throughput adaptation and the presentation and study of LoraWan performances, we noted that this research did not support the case of mobile IoT nodes. In this context, our first contribution focused on the study of the impact of mobility on the LoraWan performances based on the three most widely used mobility models, including the GaussMarkov mobility Model, the Random Waypoint Mobility Model and the Constant Position Mobility Model, using the NS3 network simulator. Simulation results for different scenarios showed that the Gauss Markov Mobility Model is more efficient compared to the other mobility models in terms of PDR and Energy.

PDR is certainly the most pertinent metric in every network, since it is ultimately desired that the rate of successfully transmitted packets will be higher. Finding the right combination aims to minimize the transmission cost and increase the rate of successfully transmitted packets. Thus, our second contribution in this thesis, as a first part, is the presentation of the spreading factor allocation scheme in the LoRaWAN network and the relationship and influence of the spreading factor on Lora performance (such as PDR, Payload, Radius, network size, ToA...) using the $NS3$ simulator and describes, examines the challenges concerning the Adaptive Data Rate (ADR) mechanism proposed by LoRaWAN as well as the different variants of this mechanism presented in the literature.

In the second part, despite the various simulations carried out and the results obtained, which support the specification [3], we consider it more appropriate to use mathematical and logical methods to validate the ADR mechanism. Therefore, we use the formal method to verify and validate the ADR protocol in the LoraWan network. We used Event-B (the formal method) to model the protocol layers and their properties, and Event-B invariants to ensure protocol consistency, and we'll be adding more guarantees to the validity of the protocol. For the moment, we're concentrating on the formal validation of the ADR mechanism on the network server side.

In a final and main contribution, we designed a security scheme based on the classes of the LoraWan module implemented under $NS-3$, and deployed the $AES-128$ bits encryption/decryption standard. We focused on the periodic sender mode as the simulation results obtained from all our contributions in this thesis. We conclude the chapter by studying the vulnerability of LoRaWAN against certain attacks. We have examined the risks of attacks and the measures that can be taken in the LoRaWAN protocol against these attacks.

The contributions presented in this thesis open up several perspectives for future work, namely in the fields of security or performance optimization. We highlight some of the more important perspectives.

In the direct continuation of our thesis work, as we have published the formal validation of ADR on the network server side, we can start the formal validation of ADR on the end device side using Event-B.

In the security section, once, we have reached the point of implementing the AES-128 bits algorithm and having results related to the PDR, ToA encryption and without encryption, energy with several packet size and CPU time, as a first step, we will try with the AES-192 bits and AES-256 bits cryptography and make a comparative study. As a second step, and based on the results obtained, prepare a survey in the application areas of each encryption mode. Now, that data encryption and confidentiality are complete, we continue to ensure data integrity and non-repudiation by implementing the AES-CMAC algorithm.

In the second step, we will explore the same scenarios, but in the context of mobile nodes with the same mobility models studied in the second chapter of this thesis.

In addition, as mentioned at the end of the fourth chapter, security issues are significantly slowing down the evolution and rapid deployment of LoRa technology and its LoRaWAN protocol. A more in-depth study of security issues is necessary to find solutions that limit attack problems, but also adapt to the limited resources of IoT objects.

Bibliography

[1] B. Zhang, V. Srinivas, Y. Zhang, and R. P. Dick, "Lora synchronized energy-efficient lp-wan," in *ICC 2023 - IEEE International Conference on Communications*, 2023, pp. 5383–5389.

[2] D. I. Săcăleanu, I. P. Manciu, and L. A. Perişoară, "Performance analysis of lora technology in wireless sensor networks," in *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 2019, pp. 1–5.

[3] Lorawan specification v1.1, Available online:. https://resources.lora-alliance.org/technical-specifications/lorawan-specification-v1-1/(accessed on 29 november 2023).

[4] Sigfox, 2019. Available online:. https://www.sigfox.com/(accessed on 24 november 2023).

[5] Narrowband internet of things (nb-iot), 2020. Available online:. https://www.gsma.com/iot/narrow-band-internet-of-things-nb-iot(accessed on 18 november 2023).

[6] M. Kalyan, V. Reddy, K. Jitesh, S. Ashif, R. K. CV *et al.*, "Lpwan technologies for iot deployment," *International Journal of Electrical Engineering and Technology*, vol. 11, no. 3, 2020.

[7] Z. A. Khan, U. Abbasi, and S. W. Kim, "Machine learning and lpwan based internet of things applications in healthcare sector during covid-19 pandemic," *Electronics*, vol. 10, no. 14, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/14/1615

[8] M. Islam, H. M. M. Jamil, S. A. Pranto, R. K. Das, A. Amin, and A. Khan, "Future industrial applications: Exploring lpwan-driven iot protocols," *Sensors*, vol. 24, no. 8, 2024. [Online]. Available: https://www.mdpi.com/1424-8220/24/8/2509

[9] Lora world coverage, Available online:. www.lora-alliance.org (accessed on 09 April 2024).

[10] J. Szewczyk, M. Nowak, P. Remlein, and A. Głowacka, "Lorawan communication implementation platforms," *International Journal of Electronics and Telecommunications*, vol. Vol. 68. No. 4, p. 841–854, 2022. [Online]. Available: http://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-a0e4fc82-0b1d-4ccb-8c2e-25aea9569117

[11] Sigfox world coverage, Available online:. www.sigfox.com/en/coverage/ (accessed on 06 April 2024).

[12] **Nouar, Abdelouahab**, M. T. Abbes, S. Boumerdassi, and M. Chaib, "Impact of mobility model on lorawan performance," *Journal of Communications*, vol. 19, no. 1, 2024.

[13] D. Patel and M. Won, "Experimental study on low power wide area networks (lpwan) for mobile internet of things," in *2017 IEEE 85th vehicular technology conference (VTC Spring)*. IEEE, 2017, pp. 1–5.

[14] K. Hu, C. Gu, and J. Chen, "Ltrack: A lora-based indoor tracking system for mobile robots," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 4, pp. 4264–4276, 2022.

[15] S. Loukil, L. C. Fourati, A. Nayyar, and C. So-In, "Investigation on security risk of lorawan: Compatibility scenarios," *IEEE Access*, vol. 10, pp. 101 825–101 843, 2022.

[16] M. Danladi and M. Baykara, "Design and implementation of temperature and humidity monitoring system using lpwan technology," *Ingénierie des Systèmes D Information*, vol. 27, pp. 521–529, 09 2022.

[17] Matt knight, "reversing lora.", Available online:. https://github.com/matt-knight/research(accessed on 22 May 2024).

[18] E. Recommendation, "Erc recommendation 70-03 relating to the use of short range devices (srd)," *Relating to the use of Short Range Devices (SRD)*, vol. 3, 2022.

[19] L. Workgroup, "Lorawan 1.1 regional parameters," *LoRa Alliance*, 2018.

[20] L. Alliance, "Lora and lorawan-a technical overview," *White paper*, 2019.

[21] Lora regional parameters, Available online:. https://www.thethingsnetwork.org/docs/lorawan/regional-parameters/(accessed on 08 september 2023).

[22] C. Lehong, B. Isong, F. Lugayizi, and A. M. Abu-Mahfouz, "A survey of lorawan adaptive data rate algorithms for possible optimization," in *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, 2020, pp. 1–9.

[23] A. Askhedkar, B. Chaudhari, M. Abdelhaq, R. Alsaqour, R. Saeed, and M. Zennaro, "Lora communication using tvws frequencies: Range and data rate," *Future Internet*, vol. 15, p. 270, 08 2023.

[24] A. Semtech, "Lora modulation basics," *Semtech Corporation, Tech. Rep.*, 2015.

[25] M. Danladi and M. Baykara, "Design and implementation of temperature and humidity monitoring system using lpwan technology," *Ingénierie des Systèmes D Information*, vol. 27, p. 521, 09 2022.

[26] L. Specification, "Lorawan specification v1.0.3," *San Francisco, CA, USA*, 2018.

[27] L. Alliance, "Lorawan specification v1. 0.2," *Date of retrieval*, vol. 13, p. 201, 2016.

[28] M. Jouhari, N. Saeed, M.-S. Alouini, and E. M. Amhoud, "A survey on scalable lorawan for massive iot: Recent advances, potentials, and challenges," *IEEE Communications Surveys & Tutorials*, 2023.

[29] L. Specification, "Lorawan specification v1. 0," *San Francisco, CA, USA*, 2015.

[30] N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. Hersent, "Lorawan specification v1.0.1," *LoRa alliance*, vol. 1, 2015.

[31] L. A. T. Committee., "Lorawan™ specification tech. rep. version 1.0.4; lora alliance technical committee: Fremont, ca, usa, 2020." 2020.

[32] A. Mahmood, E. Sisinni, L. Guntupalli, R. Rondon, S. Hassan, and M. Gidlund, "Scalability analysis of a lora network under imperfect orthogonality," 08 2018.

[33] M. A. M. Almuhaya, W. A. Jabbar, N. Sulaiman, and S. Abdulmalek, "A survey on lorawan technology: Recent trends, opportunities, simulation tools and future directions," *Electronics*, vol. 11, no. 1, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/1/164

[34] W. Ayoub, A. E. Samhat, F. Nouvel, M. Mroue, and J.-C. Prévotet, "Internet of mobile things: Overview of lorawan, dash7, and nb-iot in lpwans standards and supported mobility," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1561–1581, 2019.

[35] V. Ribeiro, R. Filho, A. Ramos, and J. Rodrigues, "Enhancing key management in lorawan with permissioned blockchain," *Sensors*, vol. 20, p. 3068, 05 2020.

[36] M. A. M. Almuhaya, W. A. Jabbar, N. Sulaiman, and S. Abdulmalek, "A survey on lorawan technology: Recent trends, opportunities, simulation tools and future directions," *Electronics*, vol. 11, no. 1, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/1/164

[37] T. Fedullo, A. Morato, G. Peserico, L. Trevisan, F. Tramarin, S. Vitturi, and L. Rovati, "An iot measurement system based on lorawan for additive manufacturing," *Sensors*, vol. 22, no. 15, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/15/5466

[38] A. Farhad and J.-Y. Pyun, "Ai-era: Artificial intelligence-empowered resource allocation for lora-enabled iot applications," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 12, pp. 11 640–11 652, 2023.

[39] H. Zhang, Y. Song, M. Yang, and Q. Jia, "Modeling and optimization of lora networks under multiple constraints," *Sensors*, vol. 23, no. 18, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/18/7783

[40] D. Magrin, M. Centenaro, and L. Vangelista, "Performance evaluation of lora networks in a smart city scenario," in *2017 IEEE International Conference on communications (ICC)*. ieee, 2017, pp. 1–7.

[41] D. Croce, M. Gucciardo, S. Mangione, G. Santaromita, and I. Tinnirello, "Impact of lora imperfect orthogonality: Analysis of link-level performance," *IEEE Communications Letters*, vol. 22, no. 4, pp. 796–799, 2018.

[42] J.-M. Kang and D.-W. Lim, "On the quasi-orthogonality of lora modulation," *IEEE Internet of Things Journal*, vol. 10, no. 14, pp. 12 366–12 378, 2023.

[43] S. Umbreen, D. Shehzad, N. Shafi, B. Khan, and U. Habib, "An energy-efficient mobility-based cluster head selection for lifetime enhancement of wireless sensor networks," *IEEE Access*, vol. 8, pp. 207 779–207 793, 2020.

[44] D. B. Johnson and D. A. Maltz, *Dynamic Source Routing in Ad Hoc Wireless Networks*. Boston, MA: Springer US, 1996, pp. 153–181. [Online]. Available: https://doi.org/10.1007/978-0-585-29603-6_5

[45] F. Geng and S. Xue, "A comparative study of mobility models in the performance evaluation of mcl," in *2013 22nd Wireless and Optical Communication Conference*. IEEE, 2013, pp. 288–292.

[46] A. Ben Yagouta, M. Jabberi, and B. Ben Gouissem, "Impact of sink mobility on quality of service performance and energy consumption in wireless sensor network with cluster based routing protocols," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, 2017, pp. 1125–1132.

[47] H. S. Guruprasad and N. Vineeth, "Performance analysis of network coded video streams in vanets based on mobility models," 06 2015.

[48] H. K. Wu, S. Nabar, and R. Poovendran, "An energy framework for the network simulator 3 (ns-3)," in *International ICST Conference on Simulation Tools and Techniques*, 2011.

[49] Network simulator (ns)-3, Available online:. https://www.nsnam.org/(accessed on 23 november 2023).

[50] B. Abdallah, S. Khriji, R. Chéour, C. Lahoud, K. Moessner, and O. Kanoun, "Improving the reliability of long-range communication against interference for non-line-of-sight conditions in industrial internet of things applications," *Applied Sciences*, vol. 14, no. 2, 2024. [Online]. Available: https://www.mdpi.com/2076-3417/14/2/868

[51] K. Anwar, T. Rahman, A. Zeb, I. Khan, M. Zareei, and C. Vargas-Rosales, "Rm-adr: Resource management adaptive data rate for mobile application in lorawan," *Sensors*, vol. 21, no. 23, p. 7980, 2021.

[52] A. Farhad, D.-H. Kim, S. Subedi, and J.-Y. Pyun, "Enhanced lorawan adaptive data rate for mobile internet of things devices," *Sensors*, vol. 20, no. 22, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/22/6466

[53] A. Farhad, D.-H. Kim, B.-H. Kim, A. F. Y. Mohammed, and J.-Y. Pyun, "Mobility-aware resource assignment to iot applications in long-range wide area networks," *IEEE Access*, vol. 8, pp. 186 111–186 124, 2020.

[54] A. Farhad, D.-H. Kim, D. Kwon, and J.-Y. Pyun, "An improved adaptive data rate for lorawan networks," in *2020 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*. IEEE, 2020, pp. 1–4.

[55] A. Farhad and J.-Y. Pyun, "Hadr: A hybrid adaptive data rate in lorawan for internet of things," *ICT Express*, vol. 8, no. 2, pp. 283–289, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405959521001788

[56] A. Farhad, G.-R. Kwon, and J.-Y. Pyun, "Mobility adaptive data rate based on kalman filter for lora-empowered iot applications," in *2023 IEEE 20th Consumer Communications Networking Conference (CCNC)*, 2023, pp. 321–324.

[57] L. Vangelista, I. Calabrese, and A. Cattapan, "Mobility classification of lorawan nodes using machine learning at network level," *Sensors*, vol. 23, no. 4, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/4/1806

[58] V. Moysiadis, T. Lagkas, V. Argyriou, A. Sarigiannidis, I. D. Moscholios, and P. Sarigiannidis, "Extending adr mechanism for lora enabled mobile end-devices," *Simulation Modelling Practice and Theory*, vol. 113, p. 102388, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X21000952

[59] J. Finnegan, R. Farrell, and S. Brown, "Analysis and enhancement of the lorawan adaptive data rate scheme," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7171–7180, 2020.

[60] M. Lodhi, L. Wang, and A. Farhad, "Nd-adr: Nondestructive adaptive data rate for lorawan internet of things," *International Journal of Communication Systems*, vol. 35, 03 2022.

[61] C. Jiang, Y. Yang, X. Chen, J. Liao, W. Song, and X. Zhang, "A new-dynamic adaptive data rate algorithm of lorawan in harsh environment," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8989–9001, 2022.

[62] F. Cuomo, D. Garlisi, A. Martino, and A. Martino, "Predicting lorawan behavior: How machine learning can help," *Computers*, vol. 9, p. 60, 07 2020.

[63] S. Cui and I. Joe, "Collision prediction for a low power wide area network using deep learning methods," *Journal of Communications and Networks*, vol. 22, no. 3, pp. 205–214, 2020.

[64] N. BENKAHLA, H. TOUNSI, Y.-Q. SONG, and M. FRIKHA, "Enhanced adr for lorawan networks with mobility," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 1–6.

[65] T. Arvind, "A comparative study of various network simulation tools," *International Journal of Computer Science & Engineering Technology*, vol. 7, no. 8, pp. 374–378, 2016.

[66] L. Campanile, M. Gribaudo, M. Iacono, F. Marulli, and M. Mastroianni, "Computer network simulation with ns-3: A systematic literature review," *Electronics*, vol. 9, p. 272, 02 2020.

[67] Z. Haider, R. Hussain, I. L. Khan, A. Shakeel, B. Ijaz, and S. A. Malik, "Evaluation of capabilities of open source cognitive radio network simulators," *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1814–1817, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:8390750

[68] A. K. Walia, A. Chhabra, and D. Sharma, "Comparative analysis of contemporary network simulators," in *Innovative Data Communication Technologies and Application*, J. S. Raj, K. Kamel, and P. Lafata, Eds. Singapore: Springer Nature Singapore, 2022, pp. 369–383.

[69] M. Almuhaya, W. Al-Areeqi, N. Sulaiman, and A. Sulaiman, *An Overview on LoRaWAN Technology Simulation Tools*, 01 2022, pp. 345–358.

[70] L. Vangelista, D. Magrin, and M. Centenaro, "Network level performances of a lora system," *Directores Lauree magistrali*, 2017.

[71] Cubecell, Available online:. https://www.heltec.org (accessed on 24 October 2023).

[72] Lorawan frequency plans, Available online:. https://www.loraantenna.com/fr/lorawan-frequency-plans-by-country-region/(accessed on 22 May 2024).

[73] C. Mostefa, T. A. Mounir, A. M. Abdelmadjid, and **Nouar, Abdelouahab**, "Ft-csma: A fine-tuned csma protocol for lora-based networks," *Journal of Communications*, vol. 19, no. 2, 2024.

[74] T. Yatagan and S. Oktug, "Smart spreading factor assignment for lorawans," in *2019 IEEE Symposium on Computers and Communications (ISCC)*, 2019, pp. 1–7.

[75] S. Maudet, G. Andrieux, R. Chevillon, and J.-F. Diouris, "Refined node energy consumption modeling in a lorawan network," *Sensors*, vol. 21, no. 19, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/19/6398

[76] M. Asad Ullah, J. Iqbal, A. Hoeller, R. D. Souza, and H. Alves, "K-means spreading factor allocation for large-scale lora networks," *Sensors*, vol. 19, no. 21, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/21/4723

[77] A. Farhad, D.-H. Kim, and J.-Y. Pyun, "Resource allocation to massive internet of things in lorawans," *Sensors*, vol. 20, p. 20, 05 2020.

[78] R. Hamdi, M. Qaraqe, and S. Althunibat, "Dynamic spreading factor assignment in lora wireless networks," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–5.

[79] S. Narieda, T. Fujii, and K. Umebayashi, "Energy constrained optimization for spreading factor allocation in lorawan," *Sensors*, vol. 20, no. 16, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/16/4417

[80] A. Loubany, S. Lahoud, and R. El Chall, "Adaptive algorithm for spreading factor selection in lorawan networks with multiple gateways," *Computer Networks*, vol. 182, p. 107491, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128620311609

[81] A. Loubany, S. Lahoud, A. E. Samhat, and M. E. Helou, "Improving energy efficiency in lorawan networks with multiple gateways," *Sensors (Basel, Switzerland)*, vol. 23, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:259125866

[82] M. S. E. Thomassen, K. S. Winkler, D. Magrin, and M. Albano, "A study of extensive lorawan downlink communication in a mobility scenario," in *Mobile and Ubiquitous Systems: Computing, Networking and Services*, S. Longfei and P. Bodhi, Eds. Cham: Springer Nature Switzerland, 2023, pp. 455–468.

[83] R. B. Sorensen, N. Razmi, J. J. Nielsen, and P. Popovski, "Analysis of lorawan uplink with multiple demodulating paths and capture effect," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.

[84] C. Caillouet, M. Heusse, and F. Rousseau, "Optimal sf allocation in lorawan considering physical capture and imperfect orthogonality," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[85] L. Amichi, M. Kaneko, N. E. Rachkidy, and A. Guitton, "Spreading factor allocation strategy for lora networks under imperfect orthogonality," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7.

[86] K. Q. Abdelfadeel, V. Cionca, and D. Pesch, "Fair adaptive data rate allocation and power control in lorawan," in *2018 IEEE 19th International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*. IEEE, 2018, pp. 14–15.

[87] A. Farhad, D.-H. Kim, and J.-Y. Pyun, "R-arm: Retransmission-assisted resource management in lorawan for the internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7347–7361, 2021.

[88] R. Kufakunesu, G. P. Hancke, and A. M. Abu-Mahfouz, "A survey on adaptive data rate optimization in lorawan: Recent solutions and major challenges," *Sensors*, vol. 20, no. 18, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/18/5044

[89] Lora alliance, Available online:. https://lora-alliance.org/in-the-news/lora-alliancer-releases-lorawanr-ts1-104-specification-simplifies-development(accessed on 09 september 2023).

[90] A. Farhad, D.-H. Kim, B.-H. Kim, A. F. Y. Mohammed, and J.-Y. Pyun, "Mobility-aware resource assignment to iot applications in long-range wide area networks," *IEEE Access*, vol. 8, pp. 186 111–186 124, 2020.

[91] N. SEMTECH, "Lorawan® mobile applications: Blind adr," 2019.

[92] A. Farhad, D.-H. Kim, J.-S. Yoon, and J.-Y. Pyun, "Feasibility study of the lorawan blind adaptive data rate," in *2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN)*, 2021, pp. 67–69.

[93] C. Mostefa, **Abdelouahab, Nouar**, T. A. Mounir, S. Boumerdassi, S. Femmam, and Z. A. Amel, "Formal validation of adr protocol in lorawan network using event-b," in *2023 7th International Conference on Computer, Software and Modeling (ICCSM)*. IEEE, 2023, pp. 11–15.

[94] H. Al-lami, "The application and analysis of a landing gear system formal model in event-b, formal model quality evaluation, the implication of godel incompleteness theory on formal systems, adaptation of axiomatic systems properties on formal models." Ph.D. dissertation, 04 2016.

[95] M. Jastram and P. M. Butler, "Rodin user's handbook: Covers rodin v. 2.8," 2014.

[96] S. Hallerstede, "Proving quicksort correct in event-b," *Electr. Notes Theor. Comput. Sci.*, vol. 259, pp. 47–65, 12 2009.

[97] J.-R. Abrial, *Modeling in Event-B - System and Software Engineering*, 01 2010.

[98] Rodin platform, Available online:. https://www.event-b.org(accessed on 29 May 2024).

[99] J. R. Abrial, "A system development process with event-b and the rodin platform," in *Formal Methods and Software Engineering: 9th International Conference on Formal Engineering Methods, ICFEM 2007, Boca Raton, FL, USA, November 14-15, 2007. Proceedings 9*. Springer, 2007, pp. 1–3.

[100] M. Butler and S. Hallerstede, "The rodin formal modelling tool," *BCS-FACS Christmas 2007 Meeting - Formal Methods In Industry, London*, 12 2007.

[101] M. J. Butler and S. Hoang, "Proceedings of the 6th rodin user and developer workshop, 2016," 2016.

[102] K.-L. Tsai, Y.-L. Huang, F.-Y. Leu, I. You, Y.-L. Huang, and C.-H. Tsai, "Aes-128 based secure low power communication for lorawan iot environments," *IEEE Access*, vol. 6, pp. 45 325–45 334, 2018.

[103] T. Perković, H. Rudeš, S. Damjanović, and A. Nakić, "Low-cost implementation of reactive jammer on lorawan network," *Electronics*, vol. 10, p. 864, 04 2021.

[104] S. Na, D. Hwang, W. Shin, and K.-H. Kim, "Scenario and countermeasure for replay attack using join request messages in lorawan," in *2017 international conference on information networking (ICOIN)*. IEEE, 2017, pp. 718–720.

[105] J. Han and J. Wang, "An enhanced key management scheme for lorawan," *Cryptography*, vol. 2, no. 4, 2018. [Online]. Available: https://www.mdpi.com/2410-387X/2/4/34

[106] S. Abboud and N. Abdoun, "Enhancing lorawan security: An advanced aes-based cryptographic approach," *IEEE Access*, 2023.

[107] I. Butun, N. Pereira, and M. Gidlund, "Analysis of lorawan v1. 1 security," in *Proceedings of the 4th ACM MobiHoc Workshop on Experiences with the Design and Implementation of Smart Objects*, 2018, pp. 1–6.

[108] A. Weinand, A. Fuente, C. Lipps, and M. Karrenbauer, "Physical layer security based key management for lorawan," 01 2021.

[109] C. El Fehri, N. Baccour, P. Berthou, and I. Kammoun, "Experimental analysis of the over-the-air activation procedure in lorawan," in *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2021, pp. 30–35.

[110] K.-L. Tsai, F.-Y. Leu, L.-L. Hung, and C.-Y. Ko, "Secure session key generation method for lorawan servers," *IEEE Access*, vol. 8, pp. 54 631–54 640, 2020.

[111] F. Hessel, L. Almon, and F. Álvarez, "Chirpotle: A framework for practical lorawan security evaluation," in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020, pp. 306–316.

[112] O. Pospisil, R. Fujdiak, K. Mikhaylov, H. Ruotsalainen, and J. Misurec, "Testbed for lorawan security: Design and validation through man-in-the-middle attacks study," *Applied Sciences*, vol. 11, no. 16, p. 7642, 2021.

[113] K.-L. Tsai, F.-Y. Leu, I. You, S.-W. Chang, S.-J. Hu, and H. Park, "Low-power aes data encryption architecture for a lorawan," *IEEE Access*, vol. 7, pp. 146 348–146 357, 2019.

[114] P. Thaenkaew, B. Quoitin, and A. Meddahi, "Evaluating the cost of beyond aes-128 lorawan security," in *2022 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2022, pp. 1–6.

[115] S. Naoui, M. E. Elhdhili, and L. A. Saidane, "Trusted third party based key management for enhancing lorawan security," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*. IEEE, 2017, pp. 1306–1313.

[116] J. Jalowiczor, J. Rozhon, and M. Voznak, "Study of the efficiency of fog computing in an optimized lorawan cloud architecture," *Sensors*, vol. 21, no. 9, p. 3159, 2021.

[117] M. Mehic, M. Duliman, N. Selimovic, and M. Voznak, "Lorawan end nodes: Security and energy efficiency analysis," *Alexandria Engineering Journal*, vol. 61, no. 11, pp. 8997–9009, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110016822001235

[118] **Nouar Abdelouahab**, M. Tahar Abbes, S. Boumerdassi, and M. Chaib, "Experimental results of using aes-128 in lorawan," *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, vol. 25, no. 5, 2025, doi: 10.17586/2226-1494-2025-25-5.

[119] Kokke, "Tiny AES in C," 2024, Accessed on: 06-11-2024. [Online]. Available: https://github.com/kokke/tiny-AES-c

[120] J. Qadir, J. E. U. Cabus, I. Butun, R. Lagerström, P. Gastaldo, and D. D. Caviglia, "Analysis of lpwan: Cyber-security vulnerabilities and privacy issues in lorawan, sigfox, and nb-iot," in *Low-Power Wide-Area Networks: Opportunities, Challenges, Risks and Threats*. Springer, 2023, pp. 139–170.

[121] E. Aras, G. S. Ramachandran, P. Lawrence, and D. Hughes, "Exploring the security vulnerabilities of lora," in *2017 3rd IEEE international conference on cybernetics (CYBCONF)*. IEEE, 2017, pp. 1–6.

[122] E. Van Es, H. Vranken, and A. Hommersom, "Denial-of-service attacks on lorawan," in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, 2018, pp. 1–6.

[123] 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.

[124] A. Mohamed, F. Wang, I. Butun, J. Qadir, R. Lagerström, P. Gastaldo, and D. D. Caviglia, "Enhancing cyber security of lorawan gateways under adversarial attacks," *Sensors*, vol. 22, no. 9, p. 3498, 2022.

[125] F. Mårlind and I. Butun, "Activation of lorawan end devices by using public key cryptography," in *2020 4th Cyber Security in Networking Conference (CSNet)*. IEEE, 2020, pp. 1–8.