TecnoLógicas
ISSN-p 0123-7799
ISSN-e 2256-5337
Vol. 22, No. 46, pp 185-194
Sep-dic de 2019

Artículo de Investigación/ Research Article

Development and Testing of a Real-Time LoRawan Sniffer Based on GNU-Radio

Desarrollo y prueba de un Sniffer en tiempo real de una red LoRawan usando GNU-Radio

Andrés F. Bravo-Montoya[1], Jefersson S. Rondón-Sanabria[2], y Elvis E. Gaona-García[3],

Recibido: 26 de julio de 2019
Aceptado: 02 de septiembre de 2019

Cómo citar / How to cite

A. F. Bravo-Montoya, J. S. Rondón-Sanabria, y E. E. Gaona-García, "Development and Testing of a Real-Time LoRawan Sniffer Based on GNU-Radio". *TecnoLógicas*, vol. 22, no. 46, pp. 185-194, 2019. https://doi.org/ 10.22430/22565337.1491

[1] Estudiante de Ingeniería Electrónica, Facultad de ingeniería, Universidad Distrital Francisco José de Caldas, Bogotá-Colombia, afbravom@correo.udistrital.edu.co
[2] Estudiante de Ingeniería Electrónica, Facultad de ingeniería, Universidad Distrital Francisco José de Caldas, Bogotá-Colombia, jsrondons@correo.udistrital.edu.co
[3] PhD. en Ingeniería, Universidad Distrital Francisco José de Caldas, Bogotá-Colombia, egaona@udistrital.edu.co

Abstract

This paper shows the vulnerabilities present in a wireless sensor network implemented over a long-range wide area network (LoRaWAN) LoRaWAN, and identifies possible attacks that could be made to the network using sniffing and/or replay. Attacks on the network were performed by implementing a protocol analyzer (Sniffer) to capture packets. The Sniffer was implemented using the RTL2832U hardware and visualized in Wireshark, through GNU-Radio. Tests showed that data availability and confidentiality could be threatened through replay attacks with LoRa server verification using HackRF One and GNU-Radio hardware. Although the LoRaWAN specification has, frame counters to avoid replay attacks, under given the right conditions, this measure could be violated even deny service to the node on the server.

Keywords

Internet of Things, Long Range Wide Area Network, Data Security, Sniffer, GNU-Radio.

Resumen

En este documento se muestran las vulnerabilidades presentes en una red de sensores inalámbricas implementada sobre una red de área amplia de largo alcance (LoRaWAN por sus siglas en inglés) LoRaWAN y se identifican los posibles ataques que se podrían realizar a la red usando sniffing y/o replay. Los ataques a la red se realizaron implementando un analizador de protocolos (Sniffer) para capturar los paquetes. El Sniffer se implementó utilizando el hardware RTL2832U y se visualizó en Wireshark, a través de GNU-Radio. Las pruebas mostraron que se pueden amenazar la disponibilidad y confidencialidad de los datos a través de ataques de replay con verificación en el LoRa server utilizando hardware HackRF One y GNU-Radio. Aunque la especificación LoRaWAN tiene contadores para evitar ataques de replay, bajo condiciones adecuadas se lograría vulnerar la red llegando a realizar la denegación del servicio del nodo en el servidor.

Palabras clave

Internet de las cosas, red de área amplia de largo alcance, seguridad de datos, Sniffer, analizador de protocolos, GNU-Radio.

1. INTRODUCTION

The concept of Internet of Things (IoT) is relatively new; it first appeared between 2008 and 2009. It was defined by CISCO Internet Business Solutions Group (IBSG) as the time when more inanimate objects were connected to the Internet than people [1]. Today, the IoT has a great impact on people's daily life. New devices with Internet connection are constantly being created, so not only people but also objects use the network in order to operate properly or perform the tasks for which they were created.

The IoT has environmental, industrial, urban, familiar, and personal applications. These new technologies offer great possibilities due to IoT's ability to capture processes and transmit information [2].

Although there are many advantages to IoT technologies, device security is a particularly important aspect, especially for wireless networks [3]. The lack of a clear and well-defined information security policy inevitably leads to unauthorized access to a network or its devices, which can cause serious problems in most cases [4].

Due to the great number of possibilities offered by the IoT, many companies have introduced innovative solutions to the market and, as a result, different infrastructures have been created for IoT management and control. Some of the solutions include SigFox, a telecommunications network with wide coverage focused on low-power devices [5]; Symphony, specialized in overcoming difficulties in LoPoWANs (Low-Power Wide Area Networks) [6]; networks, Zigbee, used in many applications due to its short range, low power consumption, low data transmission, and high security [7]; and LoRaWANs, a scheme for addressing long-range links also known as LoRa [8]. This set of standards includes new technological approaches to data transmission security [9]. In addition to the development of infrastructure for IoT management, several technologies are widely used in the deployment and success of IoT-based products and services: radio frequency identification (RFID), wireless sensor networks (WSN), middleware, cloud computing, and IoT software applications [10]. Wireless technologies change very rapidly; new products and features are continuously introduced. However, their new capabilities can produce new threats or vulnerabilities to equipment and data security.

Networking with LoRa devices can be divided into two fundamental parts: one section from the end nodes to the gateway, and another section from the gateway to the servers. On the one hand, the second section includes several available security solutions, as this is not unique to LoRa devices. On the other hand, the first section may be susceptible to security attacks, so this part of the network cannot be considered a trusted network entity [11].

When analyzing the vulnerabilities in the first section of the network, the possibility of making attacks to the network by using sniffing and/or relay techniques was found. From these possible attacks, it is clear that LoRa has inherent weaknesses caused by the compromises made in its design [11].

As mentioned above, the main risk of these security failures is the theft of sensitive and/or confidential information; therefore, new methods are necessary to eliminate or minimize these failures [12].

For this purpose, one of the most practical tools in network security is a sniffer. This tool is usually employed by hackers but also network managers to maintain the security level of their networks (identifying the vulnerabilities it may present, such as device A cannot establish communication with device B) and even to efficiently manage the network, since it can identify the stability

of a network with tremendous ease and perform audits in a very short time.

A sniffer records all the information that is sent in a wireless network, as well as any activity carried out. That is, it has the ability to capture and record any transfer of information, through which it is possible to discover bottlenecks in the network [13].

This article is organized as follows: Section 2 describes the configuration and hardware used for testing. Section 3 reports the test results of network attacks and the discussion. Finally, Section 4 presents the conclusions and future research.

2. METHODOLOGY

LoRaWAN defines two node activation procedures: 1) Over The Air Activation (OTAA), in which the final device sends a join request to the gateway and the gateway returns the data from the network server; and 2) Activation by Personalization (ABP), in which the required information is stored in the memory of the nodes, so communication is not necessary to join the network.

In addition, LoRaWAN networks have two security layers: the "*Network Session Key*", which ensures the authenticity of the node; and the "*Application Session Key*", used for data reliability [14]. However, in order to test the security provided by the protocol, some attacks were carried out on an A-class network. The measurements were taken with the hardware listed below:

-MultiTech mDot nodes
-LM35 temperature sensors
-Pressure and temperature sensor BMP280
-RTL2832U
-HackRF One analyzer
-Wireshark software
-GNU-Radio
-MultiTech gateway

The network was implemented with the previous hardware plus a network server that was also a LoRa server.

2.1 Eavesdropping

This kind of attack requires a sniffer to capture data passively; in this work packets sent from nodes to the gateway are captured by using GNU-Radio, Wireshark, and RLT2832U. Fig. 1, shows the configuration used to implement the eavesdropping test. The four A-class nodes are configured in a star topology, in which the gateway and the sniffer receive the sent data.

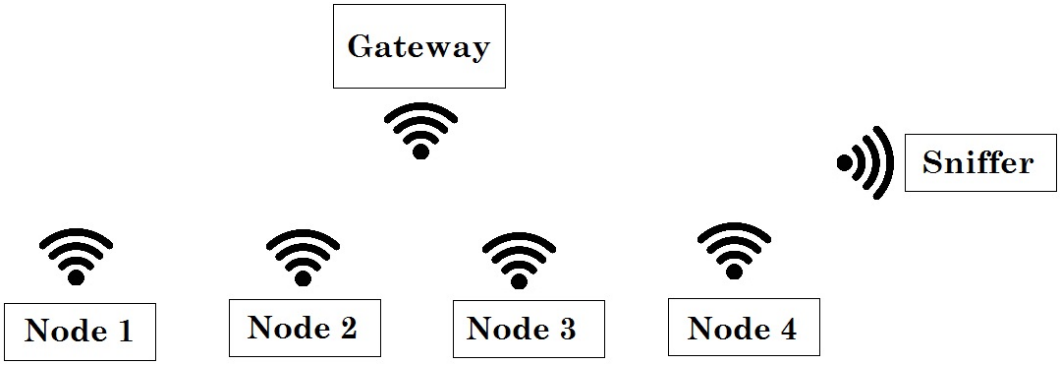

Fig. 1. Network Topology. Source: Authors.

A simple configuration in GNU-Radio is used to capture and send the data received by the sniffer. Fig. 2, shows this arrangement with a Spread Factor (SF) of about only 12.

LoRa Receiver blocks were obtained using the gr-lora repository [15]. The RTL-SDR Source block enables the configuration of the hardware parameters of the RTL2832U sniffer (sample rate, center capture rate, and gain, among others). WX GUI blocks are employed to display the captured signal (Fast Fourier Transform and the signal spectrogram). LoRa Receiver blocks capture the frames received by the sniffer in a specific channel; in these blocks, it is possible to configure the central frequency, the reception channel, the signal bandwidth, and the propagation factor. Three blocks are used to capture the three possible channels in a transmission and, finally, the Message Socket Sink blocks allow GNU-Radio to communicate with Wireshark software and send the data captured to the port and IP address specified in the block as well as the LoRaWAN frame counter.

2.2 Replay

This attack, which consists in copying the transmitted signal to supplant the node, was implemented with a Software-Defined Radio (SDR) device. The device, a HackRF One, was configured through GNU-Radio, allowing it to make copies (Fig. 3) of the signals transmitted by the nodes to send them later (Fig. 4).

The replay tests were performed with three configurations. The first one verified if the server accepted the data copied from the node. The second one copied the data without the join request. The third one used the frame counter.

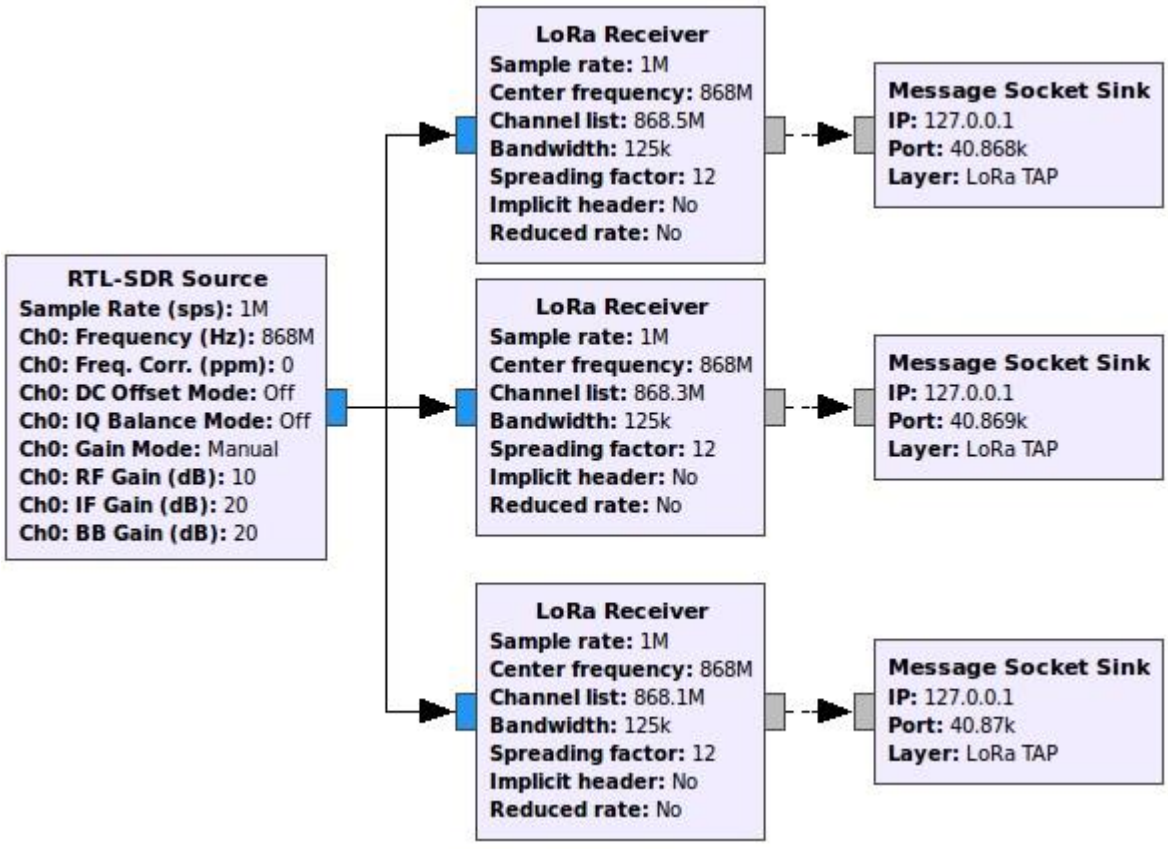

Fig. 2. GNU-Radio blocks. Source: Authors.

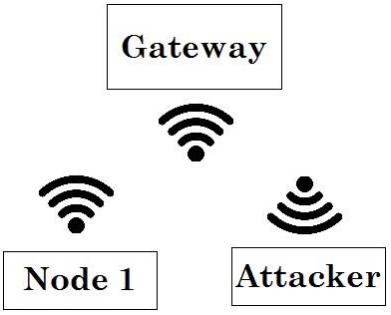

Fig. 3. General RX configuration. Source: authors.

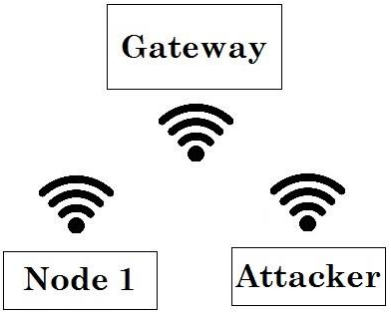

Fig. 4. General TX configuration. Source: Authors.

2.2.1 Configuration 1

For this scenario, a malicious node was used to access the network by copying authentication data and some data messages collected by sensors in order to verify if the server accepted the copied messages. After copying the signals with the HackRF One, the authentic node was disconnected from the network and the signal replicas of the malicious node continued to be sent.

2.2.2 Configuration 2

In this test, the data was copied without the join-request, keeping the node connected in order to falsify the data.

2.2.3 Configuration 3

This configuration consisted in waiting until the sequential number of the frame-counter inside the message was restarted

The test was performed while waiting for the counter to be reset to zero in order to send the copied data at the time the counter matches the counter of previously copied messages.

3. RESULTS AND DISCUSSION

The following are the results obtained after carrying out the attacks described above and a discussion about the possible causes that enabled them.

3.1 Eavesdropping

Fig. 5 shows the data captured in Wireshark, in which the received data that belongs to a specific node is discriminated. The captured data can be seen in hexadecimal format (but they are encrypted), and the data organized in the blocks sent by a UDP frame are located at the bottom.

Evidently, the attack was successful. However, the implemented sniffer did not inform Wireshark of the response messages from the server to the nodes. In addition, the counter was not encrypted.

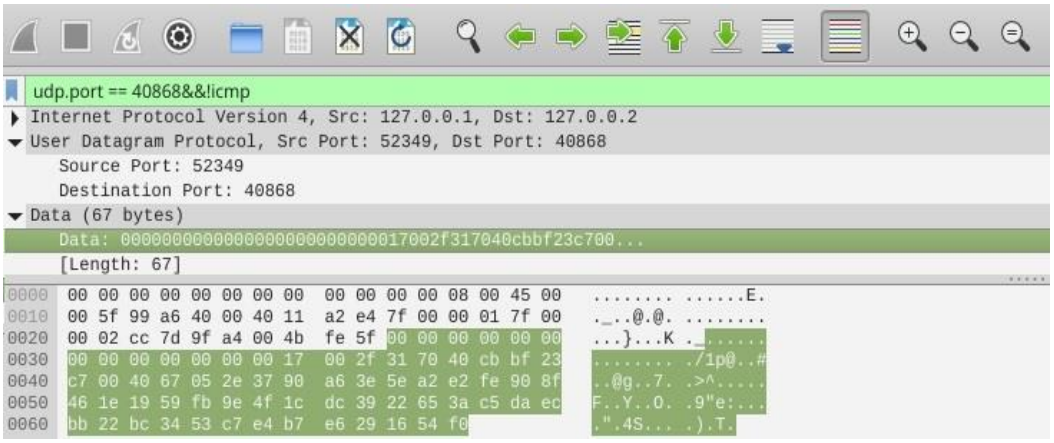

Fig. 5. Captured data. Source: Authors.

3.2 Replay

The results obtained with the different configurations described above are the following.

3.2.1 Configuration 1

In order to verify whether the attack was successful, the application response of a message sent by a genuine node was compared to the application response of a message sent by the malicious node. Fig. 6 and Fig. 7 show the application responses to the messages.

Request messages in the node and answer messages in the server are necessary. They are captured by the malicious device that copies the signals. However, the answer messages are not relevant to the proposed test since, when the attack is continuously performed, the request and answer messages are sent again by the malicious node, so the attack fails. Although the response messages from the server could be removed from the copied file, the attack would still fail because the malicious device is unable to obtain the authentication keys sent in response to the forged join request.

3.2.2 Configuration 2

Although the data sent by the malicious node is received at the gateway and its identifier suggests that it was sent by the original node, the server rejects the data and they are not considered due to a synchronism failure in the message counter field. Whether the authentic node continues to send messages or not, the counter in the malicious node will not match the number stored in the counter that controls the server.

3.2.3 Configuration 3

The success or failure of the attack is verified by comparing the responses of the application to the different messages that were sent. Fig. 8, shows the synchronism of both the malicious node and the authentic node with the counter. In addition, the malicious node sends the message before the authentic node does.

Fig. 9, shows that the server only accepts messages from the authentic node. Activation by ABP has a critical vulnerability because the keys are invariable and do not need constant authentication in the network. Therefore, a malicious message, as long as it meets the following requirements, can be accepted by the LoRaWAN network server: -Session keys are the same as those of an accepted end device.

-DevAddr field is the same as that of an accepted end device.

UPLINK	12:28:54 PM	UnconfirmedDataUp	c7202c4f
UPLINK	12:28:49 PM	UnconfirmedDataUp	c7202c4f
DOWNLINK	12:28:46 PM	JoinAccept	
UPLINK	12:28:45 PM	JoinRequest	008000000000d638

Fig. 6. Messages sent by the real node. Source: Authors.

UPLINK	11:53:09 AM	JoinRequest	70b3d58ff0030829
UPLINK	11:53:07 AM	JoinAccept	
UPLINK	11:53:02 AM	JoinRequest	70b3d58ff0030829

Fig. 7. Messages sent by the malicious node. Source: Authors.

Fig. 8. Gateway message reception. Source: Authors.

Fig. 9. Server reception. Source: Authors.

-Frame counter value is acceptable [16].
-Session keys are the same as those of an accepted end device.
-DevAddr field is the same as that of an accepted end device.
-Frame counter value is acceptable [16].

Although the previous conditions were met, the attack was not successful because the initialization of the nodes was carried out by OTAA. This created new coding keys each time a new session was completed. In addition, the malicious device (HackRF One) could not accurately copy the signal due to interference factors, sampling rate, or message transmission power. As a result, the message format was not valid for the application and rejected despite the fact that it was synchronous, i.e., it matched the value of the counter.

4. CONCLUSIONS

The main problem in carrying out an eavesdropping attack in a LoRaWAN configuration is obtaining the required hardware because GNU-Radio configures the hardware and manipulates the signals with relatively effortless simplicity.

A sequence of frames was captured, visualized, and analyzed during the eavesdropping attack.

The implemented sniffer captures messages sent from the node to the server, but not the responses from the server to the node when performing replay attacks. As noted, the HackRF One successfully copied all the messages. By capturing the response message from the server to the node, it will therefore decode the subsequent messages sent by the node.

Said sniffer only takes the message if it is configured with the same SF as the message sent, which hinders the attack on the node if this parameter is set in the message sent.

Frame counters are a measure proposed by the LoRaWAN specification to avoid replay attacks. However, given the right conditions, this measure could be violated. Even if the malicious node manages to keep sending frames for enough time, a denial of service to the node by the server could occur.

It was determined that a LoRaWAN implements has implements good measures to avoid replay attacks in OTAA activation. ABP activation countermeasures were not tested since that type of activation was not used.

5. ACKNOWLEDGMENTS

This work was supported by the CIDC (Centro de Investigación y Desarrollo Científico) of Universidad Distrital Francisco José de Caldas, Bogotá, Colombia, and the research group GITUD.

6. REFERENCES

[1] M. H. Miraz, M. Ali, P. S. Excell, and R. Picking, "A review on Internet of Things (IoT), Internet of Everything (IoE) and Internet of Nano Things (IoNT)," in *2015 Internet Technologies and Applications (ITA)*, Wrexham, 2015, pp. 219–224. https://doi.org/10.1109/ITechA.2015.7317398

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Futur. Gener. Comput. Syst.* vol. 29, no. 7, pp. 1645–1660, Sep. 2013. https://doi.org/10.1016/j.future.2013.01.010

[3] M. Shin, J. Ma, A. Mishra, and W. A. Arbaugh, "Wireless Network Security and Interworking," *Proc. IEEE*, vol. 94, no. 2, pp. 455–466, Feb. 2006. https://doi.org/10.1109/JPROC.2005.862322

[4] J. Botero Valencia, L. Castaño Londoño, and D. Marquez Viloria, "Trends in the Internet of Things," *TecnoLógicas*, vol. 22, no. 44, pp. I–II, Jan. 2019. https://doi.org/10.22430/22565337.1241

[5] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express*, vol. 5, no. 1, pp. 1–7, Mar. 2019. https://doi.org/10.1016/j.icte.2017.12.005

[6] B. Singh and B. Kaur, "Comparative study of

Internet of Things infrastructure and security," in *Global Wireless Submit 2016*, Aarhus Denmark, 2016.

[7] S. Serna, "Especificación de Perfil Zigbee para Monitoreo y Control de Plantas Industriales," *TecnoLógicas*, no. 23, pp. 167-185, Dec. 2009. https://doi.org/10.22430/22565337.238

[8] Q. Zhou, K. Zheng, L. Hou, J. Xing, and R. Xu, "Design and Implementation of Open LoRa for IoT," *IEEE Access*, vol. 7, pp. 100649–100657, Jul. 2019. https://doi.org/10.1109/ACCESS.2019.2930243

[9] S. Cruz-Duarte, P. A. Gaona-Garcia, and E. E. Gaona-Garcia, "Cybersecurity In Microgrids," in *2018 6th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, Barcelona, 2018. pp. 7–12. https://doi.org/10.1109/W-FiCloud.2018.00008

[10] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Bus. Horiz.*, vol. 58, no. 4, pp. 431–440, Jul. 2015. https://doi.org/10.1016/j.bushor.2015.03.008

[11] E. van Es, "LoRaWAN vulnerability analysis:(in) validation of possible vulnerabilities in the LoRaWAN protocol specification.," Tesis Maestría, Open University of the Netherlands, 2018.

[12] 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)*, Exeter, 2017, pp. 1- 6. https://doi.org/10.1109/CYBConf.2017.7985777

[13] M. A. Qadeer, A. Iqbal, M. Zahid, and M. R. Siddiqui, "Network Traffic Analysis and Intrusion Detection Using Packet Sniffer," in *2010 Second International Conference on Communication Software and Networks*, Singapore, 2010, pp.313–317. https://doi.org/10.1109/ICCSN.2010.104

[14] "LoRa Alliance," *LoRaWAN™ Specification V1.1*, 2015. [En línea] Disponible en: https://lora-alliance.org/resource-hub/lorawanr-specification-v11

[15] Github, "GitHub - rpp0/gr-lora: GNU Radio blocks for receiving LoRa modulated radio messages using SDR," *GitHub - rpp0/gr-lora: GNU Radio blocks for receiving LoRa modulated radio messages using SDR*, 2019. [En línea] Disponible en: https://github.com/rpp0/gr-lora

[16] X. Yang, "LoRaWAN: Vulnerability Analysis and Practical Exploitation," Tesis Maestría, Delft University of Technology, 2017. http://resolver.tudelft.nl/uuid:87730790-6166-4424-9d82-8fe815733f1e