A Small LoRaWAN Network

Armstrong, Samuel E.
University of Kentucky
(Dated: September 20, 2021)

Abstract - Long Range Wide Area Network (LoRaWAN) is an emerging technology that uses low-cost, low-energy intermediate gateways between a central network and end devices. Starting in 2009, LoRaWAN has gained popularity and has been deployed in many environments including agriculture, healthcare, and cities. With the tremendous amount of data that is being generated today, LoRaWAN can distribute network traffic and manage networks efficiently by converting radio frequency (RF) packets from various sensors to internet protocol (IP) packets bidirectionally. LoRaWAN has already proven to be a beneficial resource when managing data and providing a low-cost alternative to conventional network computations.

I. INTRODUCTION

One of the challenges of working with LoRaWAN is understanding how the underlying protocol works. LoRaWAN networks are made up of three device categories: end devices, gateways, and servers.

First, end devices can range from GPS trackers to door sensors to soil moisture sensors with many more options in between. These devices, at least in the United States, operate on the 915 MHz radio band. For comparison, WiFi networks usually operate on the 2.4 or 5.0 GHz bands, which is around 3 to 5 times greater than frequency of LoRaWAN. This means that the radio frequency (RF) packets produced on these devices have the ability to travel greater distances based on the longer wavelength. End devices are primarily data collectors that create and send RF packets to gateways. This is usually referred to as an uplink message.

Next, gateways are used as intermediary points between end devices and servers. Their main use is to convert RF packets transmitted by end devices to internet protocol (IP) packets that can be sent via the internet to a server. Gateways also have the ability to form and send RF packets back to end devices, usually referred to as downlink messages. This messages are usually formed on the server and sent to the gateway for transmission to the specified end device.

Finally, a servers main purpose is to receive and decrypt IP packets from gateways and provide data aggregation on information being collected by end devices. Servers also handle the acceptance or denial of devices on a network. This is a crucial exchange between devices and the server because end devices will not start collecting data if they are denied, unacknowledged, or sent a malformed acceptance message. If a device is accepted, a "join-accept" message is sent to the specific gateway for transmission to the end device.

A typical architecture for a LoRaWAN network can be found in Figure 1.

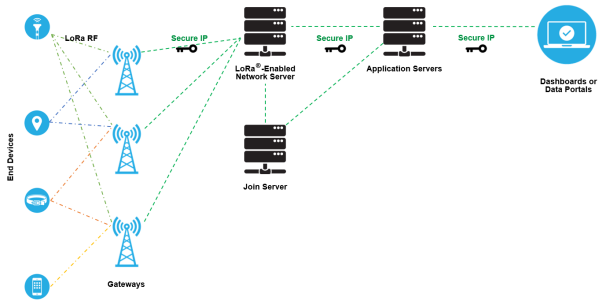

FIG. 1. A Typical LoRaWAN Network [7]

A. Alternative to Creating a LoRaWAN Network from Scratch

While the LoRaWAN specification offers solutions to problems that many developers and non-developers have, building a network from scratch would not be feasible or worthwhile for small projects. Most developers turn to third-party services to process their data. By far, The Things Network (TTN) is the most widely used service for organizing gateways and end devices. TTN is an open-source, decentralized platform for developers to view and forward data produced from end devices and gateways [3]. For newcomers, TTN is a great resource that provides a quick and easy way to start collecting LoRaWAN device data.

However, this service doesn't provide the customizability that some developers need. For example, TTN provides users with the option to place code directly within the site to decrypt end device payloads. This is helpful but does not provide the level of data processing and manipulation that some developers require, such as connecting a database. There are plugins that a developer can connect that offer a higher level of manipulation, but setting this up complicates matters further. This might turn developers away from TTN or LoRaWAN in general.

II. PROJECT DESIGN

In this project, I aim to create an open-source application that makes it easier for developers to connect, process, and store data created in a LoRaWAN network. To accomplish this task, I will use a LoRaWAN GPS tracker, a door sensor, and create a LoRaWAN gateway, which will act as a miniature LoRaWAN network to sufficiently test the hardware and software applications that I have outlined below.

A. Hardware

For hardware, this project will use a Raspberry Pi, a Dragino LoRaWAN Concentrator, a Dragino GPS tracker, and a Dragino door sensor (outlined in *Hardware Specification*).

To create a gateway, the Dragino LoRaWAN concentrator must be connected to the GPIO headers on the Raspberry Pi. This operation is simple if following the included instructions provided online by Dragino. After attaching the concentrator to the Raspberry Pi, a packet forwarding service will need to be installed. The purpose of a packet forwarding service is to facilitate the conversion of RF to IP packets and transfer IP packets from gateways to a server. This project uses an open-source packet forwarding service from Semtech, which can be found on their Github page *here*. For this project, the gateway and server will operate side-by-side on the Raspberry Pi using loopback on port 1700. This gateway will be stationed on the second floor of a building near a window, which will reduce the maximum range of the network. However, this should not cause any significant loss of packets from the short ranges (<1 km) that will be tested in this project.

The Dragino GPS Tracker is a self-contained device that does not require setting up out of the box. The tracker is equipped with a lithium-ion battery that Dragino claims to last 19 days using default settings and can be charged with a micro-USB connection. Once charged and powered on, the tracker will start to search for satellites. If the device is placed outside in clear view of the sky, it usually will not take more than 30 seconds to acquire a fix. However, by bringing the device indoors, it is likely impossible for it to acquire a fix. If leaving the device outdoors is not an option, there are a multitude of "come to attention" or "AT" commands that can be issued to the device (via a serial connection) to boost signal strength and change the search mode of the device as well as many other adjustments. This device will be stationary and placed outdoors for preliminary data collection, but will be eventually carried around short distances after the software has been created and tested.

The Dragino door sensor is similar to the GPS tracker as it is ready to connect to the network out of the box. However, this device is powered by a CR2032 coin battery. Dragino claims that this device can transmit up to 12,000 uplinks (usually door open or closed) messages in optimal conditions. This device also can connect to a computer using a serial connection to change parameters using AT commands. This device will be mounted on the entrance of the building that the gateway is based.

B. Software

The main software components that are necessary for this project are:

- a packet forwarding service (as described above) for the gateway (Semtech)
- a program to receive and transmit data to and from end devices connected the server
- a program to decrypt and encrypt incoming and outgoing messages
- a program to manipulate decrypted data into a human-readable format
- a DBMS
- a RESTful API

Once an end device is powered on, it will immediately transmit a join request, which contains device-specific information, to any LoRaWAN gateway that is listening. This is where a program to listen for new packets is necessary. When a gateway receives this request it can accept or deny the device on the network. If accepted, the server will create an Application Session Key (AppSKey) and a Network Session Key (NwkSKey) using the device-specific data received and a nonce that will be used to encrypt and decrypt further communications between the device. These keys and payload contents will then be stored in a MySQL database for future use. Next, when a user needs to view the payload, a decryption and encryption program will need to be used. After this data has been decrypted it will need to be processed so the payload is human readable. This will be done using the payload decrypter described in the user manual for each end device. Last, a RESTful API will be used to query the database for sensor data.

Security is also a concern of this application. Luckily, LoRaWAN networks can be secured with end-to-end encryption using AES-128, which will be used in this project. Also, by using a NwkSKey between an end device and the server to prevent message tampering and an AppSKey between the end device and the application messages will be verifiable and only decryptable by authorized parties.

III. HARDWARE SPECIFICATION

There are four hardware components necessary to complete this project:

- ***Raspberry Pi 3 Model B+***
- ***Dragino LoRaWAN Concentrator (PG1301-915)***
- ***Dragino GPS Tracker (LGT-92-LI-US915)***
- ***Dragino Door Sensor (LDS01-US915)***

The foundational component of this project is the Raspberry Pi. This device is equipped with a 64-bit quad-core ARM processor, 1GB of LPDDR2 SDRAM, 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, and an extended 40-pin GPIO header [2]. These features provide the most flexibility and computational power to accomplish the tasks outlined in the *Design* section. The operating system (OS) used is Raspbian, which is based on the Debian OS and has been optimized specifically for Raspberry Pi's [5]. This OS provides the functionality necessary to complete the tasks in the *Objectives* section.

The primary LoRaWAN component is the concentrator. Combined with the Raspberry Pi, the LoRaWAN Concentrator can transmit packets to and receive packets from end devices to create a gateway. Dragino claims this device can handle 5,000 end devices per km^2 area [1] and according to Semtech, a LoRaWAN research and development company, using a bitrate of 5470 bps can achieve a max range of 2 km (in low interference environments). This means the theoretical area that could be covered by a single gateway is approximately 12.5 km^2 [8].

IV. LORAWAN SPECIFICATION

As mentioned above, LoRaWAN is end-to-end encrypted with payloads consisting of many parts. Encrypting and decrypting these payloads correctly is crucial to the success of the network.

To start, each end device (once powered on) will send an unencrypted join-request packet to the server. This is the only packet that is not encrypted across the devices' uptime. This request consists of an Application Extended Unique Identifier (AppEUI, first 8 bytes), a Device Extended Unique Identifier (DevEUI, next 8 bytes), a Device Nonce (DevNonce, next 2 bytes), and a calculated Message Integrity Code (MIC, last 4 bytes). This gets processed on the server and a session key and join-accept packet is transmitted back to the device. After the device receives this, it will start to send encrypted sensor data to the server. See Figure 2.

Next, sensor data payloads contain fields as described in Figure 3. Similar to the physical and data link layers of the OSI model, a data payload contains a header and error correcting/detecting information. When this payload reaches the server it will still be encoded with base64 and encrypted with AES 128. The most important part of this string is the frame payload (FRM payload), which contains the actual data produced by the sensor. After

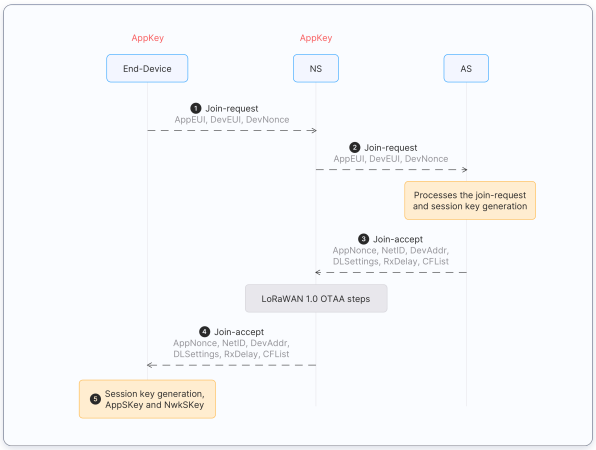

FIG. 2. End Device Activation [4]

converting base64 to hex and using the message counter (provided in the payload), the AppSKey, and the device address it is possible to decrypt the FRM payload. This will only receive the encrypted MAC payload. The length of this data changes from device to device, therefore it is necessary to check device documentation for specifics when decoding data. For example, the GPS tracker that is used in this project has a FRM payload of 18 bytes, which most importantly includes latitude, longitude, altitude, and battery voltage information.

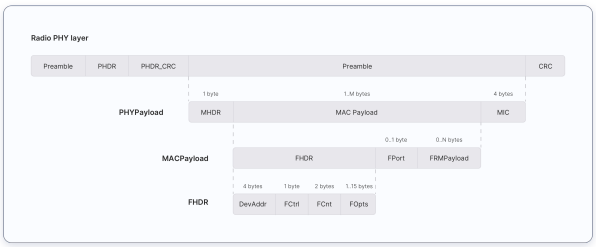

FIG. 3. Payload Description [6]

V. OBJECTIVES

The goal of this project is to learn, understand, and develop a small LoRaWAN network. To accomplish this I will create a simplified version of The Things Network with added customizability. The three steps to complete this task are:

1. Create a back-end application to handle receiving, transmitting, and processing sensor data.
2. Create a RESTful API to view sensor data
3. Test the range of devices

REFERENCES

[1] *10 channels - LoRaWAN GPS Concentrator for Raspberry Pi*. Dragino. URL: https://www.dragino.com/products/lora/item/149-lora-gps-hat.html (visited on 08/30/2021).

[2] *Buy a Raspberry Pi 3 Model B+*. Raspberry Pi, 2018. URL: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/ (visited on 08/30/2021).

[3] *Devices — The Things Network*. The Things Network. URL: https://www.thethingsnetwork.org/docs/devices/ (visited on 09/06/2021).

[4] *End Device Activation*. The Things Network. URL: https://www.thethingsnetwork.org/docs/lorawan/end-device-activation/ (visited on 09/13/2021).

[5] *FrontPage - Raspbian*. Raspbian. URL: https://www.raspbian.org/ (visited on 08/30/2021).

[6] *Message Types*. The Things Network. URL: https://www.thethingsnetwork.org/docs/lorawan/message-types/ (visited on 09/13/2021).

[7] *Typial LoRaWAN network implementation*. Semtech. URL: https://lora-developers.semtech.com/library/tech-papers-and-guides/lora-and-lorawan/ (visited on 09/03/2021).

[8] *What are LoRa® and LoRaWAN®?* Semtech. URL: https://lora-developers.semtech.com/library/tech-papers-and-guides/lora-and-lorawan/ (visited on 08/30/2021).