Asset Tracking Using LoRaWAN: Experiments Concerning Effective Range and Signal Interception

Frank T. Johnsen[a] and Torkjel Søndrol[a]

[a]Norwegian Defence Research Establishment (FFI), Kjeller, Norway

ABSTRACT

The Internet of Things (IoT) is growing steadily in the civilian commercial sector. One can expect that some of the work done here can potentially be used for military purposes as well. Some use cases, e.g., logistics and health care applications, are typically applicable in both civilian and military settings. Considering logistics, one application that can possibly benefit from using IoT is that of asset tracking. Potentially, we can achieve reliable, affordable and long range position reporting using contemporary consumer electronics. This technology capability may enable new possibilities for the future of command and control, which should harness such novel capabilities as they become available.

This paper covers our experiments with Commercial Off-The-Shelf (COTS) IoT products, and using these for asset tracking over the Long Range Wide Area Network (LoRaWAN) protocol. We evaluate the setup from two perspectives: that of effective range, and that of signal interception.

Keywords: LoRaWAN, asset tracking, security

Topic 6: Experimentation, Analysis, Assessment and Metrics

Paper ID 11

Point of contact

Frank T. Johnsen
Norwegian Defence Research Establishment (FFI)
P.O. Box 25, 2027 Kjeller, Norway
E-mail: Frank-Trethan.Johnsen@ffi.no

1. INTRODUCTION

A civilian trend which is gaining traction also for military use is the deployment of cheap sensor systems as a means to augment the information already available to military decision makers today.[1] This civilian trend, known as the Internet of Things (IoT), can be defined as follows:[2]

> "IoT describes the revolution already under way that is seeing a growing number of Internet enabled devices that can network and communicate with each other and with other web-enabled gadgets. IoT refers to a state where Things (e.g., objects, environments, vehicles and clothing) will have more and more information associated with them and may have the ability to sense, communicate, network and produce new information, becoming an integral part of the Internet. A widespread Internet of Things has the potential to transform how we live in our cities, how we move, how we develop sustainably, how we age, and more."

Work by Suri et al.,[1] in addition to promoting benefits such as low cost, further highlights technical barriers that may hinder military adoption of IoT, including issues related to network utilization and interoperability. Network resources are often scarce in tactical military scenarios. There is a need to prioritize which types of information are allowed to consume network resources in the radio based network systems that the forces use for mission critical tasks, such as coordination and position reporting. Enter the possibilities of leveraging IoT; here a whole new range of devices and communication protocols are being introduced. If we are able to use these new technologies, we may be able to offload the traditional, narrow communications channels of the tactical networks for certain specific applications. An interesting feature of the Long Range Wide Area Network (LoRaWAN) is that it is feasible to obtain and deploy this protocol stack as a separate, private infrastructure.

In this paper we investigate asset tracking using Commercial Off-The-Shelf (COTS) products as our data sources. We use these to provide timestamped Global Positioning System (GPS) positions over LoRaWAN. The Long-range Radio (LoRa) protocol, which is a relatively new, promising development for civilian IoT communications is the radio layer of LoRaWAN. It has been proposed as a carrier for military applications in the Internet of Battlefield Things (IoBT).[3] As such, this protocol is also interesting as a new technology supporting the future command and control (C2) systems. Using a COTS rugged LoRaWAN/4G gateway to interconnect these data sources, we use the lightweight industry standard Message Queuing Telemetry Transport (MQTT) publish/subscribe protocol to further disseminate data from the gateway to the cloud.

This paper covers the experiments we performed with the above mentioned hardware and software. The work presented here continues the initial efforts documented in,[4] where we examined the LoRa protocol. We had three goals with these experiments:

1. Evaluate COTS IoT products and protocols,
2. see how LoRaWAN fares as the underlying protocol for asset tracking, and
3. investigate the feasibility of intercepting the communication.

The motivation behind these goals is that we can potentially use COTS IoT products as low-cost supplements (or even replacements) for existing applications. Conversely, we may also be able to leverage IoT as an entirely new source of information, augmenting existing situation information. We focus on two specific aspects of LoRaWAN in this paper; that of the effective range, which is important to know about if considering to use the protocol for asset tracking. Further, we consider particularly one aspect of cyber security, that of information interception. If we are going to consider military applications of these civilian COTS IoT products, we also need to know about the potential risks of an adversary detecting the signal and eavesdropping on the communications. Therefore, using COTS approaches, we investigate detecting and intercepting LoRaWAN traffic.

The remainder of this paper is organized as follows: The technologies behind the LoRa and LoRaWAN protocols are further discussed in Section 2. Section 3 presents the hardware and software components used in our experiments. Sections 4 and 5 present our range and security experiments. Related work is discussed in Section 6. Section 7 concludes the paper. Finally, Section 8 outlines future work.

2. AN INTRODUCTION TO LORA AND LORAWAN

LoRa is a proprietary, closed source wireless protocol developed by Cycleo before being acquired by Semtech in 2012.[5] It is designed for low-powered devices and uses a Chirp Spread Spectrum (CSS)[6] wide band waveform, making it very suitable for allowing IoT devices to communicate over long ranges. In addition to this, the LoRa Alliance defines a higher level media access control (MAC) layer on top of LoRa, and together these form the complete LoRaWAN protocol stack as shown in Figure 1.

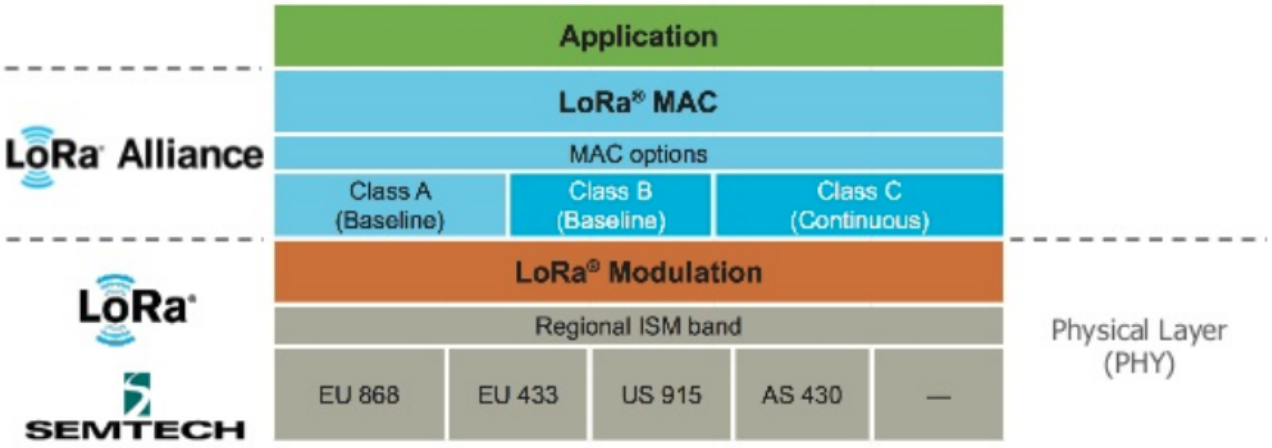


Figure 1: The LoRa and LoRaWAN protocol layers.[7]

As shown in Figure 1, LoRa is designed to operate within the unlicensed Industrial, Scientific and Medical (ISM) band, thus operating at 868 MHz in Europe and 915 MHz in the USA (for an overview of frequency bands that LoRaWAN can use in different regions, see[8]). There are also different spreading factors (SF) and bandwidths (BW) defined for the different regions in order to ensure fairness among all competing users.[9] The sensors used during our experiments used the 868 MHz band with SF7 and BW125KHz.

A deployment of LoRaWAN has an architecture that consists of certain elements: The devices (IoT products sending data), a concentrator/gateway that typically bridges the LoRaWAN domain to the Internet, and finally an application server that is the end recipient of the data. The entities, information exchange and security measures are illustrated in Figure 2.

As LoRa is a pure modulation protocol, security and addressing is performed on the MAC layer of LoRaWAN. Each LoRaWAN device is assigned two extended unique identifiers (EUIs). The first is a 64-bit device identifier (DevEUI) which is set by the device vendor. The second is a 64-bit application identifier (AppEUI) that is set by the application provider. LoRaWAN frames are both integrity and confidentiality protected, as shown in Figure 3. First the payload is encrypted using AES128-CTR using an application key (AppSKey). Then a frame counter is added to the header, and the AES-CMAC of the entire frame is calculated using a network key (NwkSkey) and appended to the frame.[10] NwkSkey is made available to the LoRaWAN network server, while AppSKey is available to the application server. This ensures that the network server is able to verify the integrity of messages, but cannot read the actual payload as Figure 2 shows.

There are two protocols available for authenticating the device with the gateway and establishing AppSKey and NwkSKey. The first is called Activation by Personalization (ABP), where both keys are statically assigned to the device and remain the same until they are manually changed. This creates challenges when it comes to proper key management, but makes the device available on the network immediately after being powered on, since key negotiation is unnecessary. The second protocol is called Over The Air Activation (OTAA) and is recommended as opposed to ABP. Here, both the device and backend are provided with a static key called AppKey, which is used to encrypt the key agreement protocol of AppSKey and NwkSKey using AES128. This way both keys are changed each time a device is registered.[13]

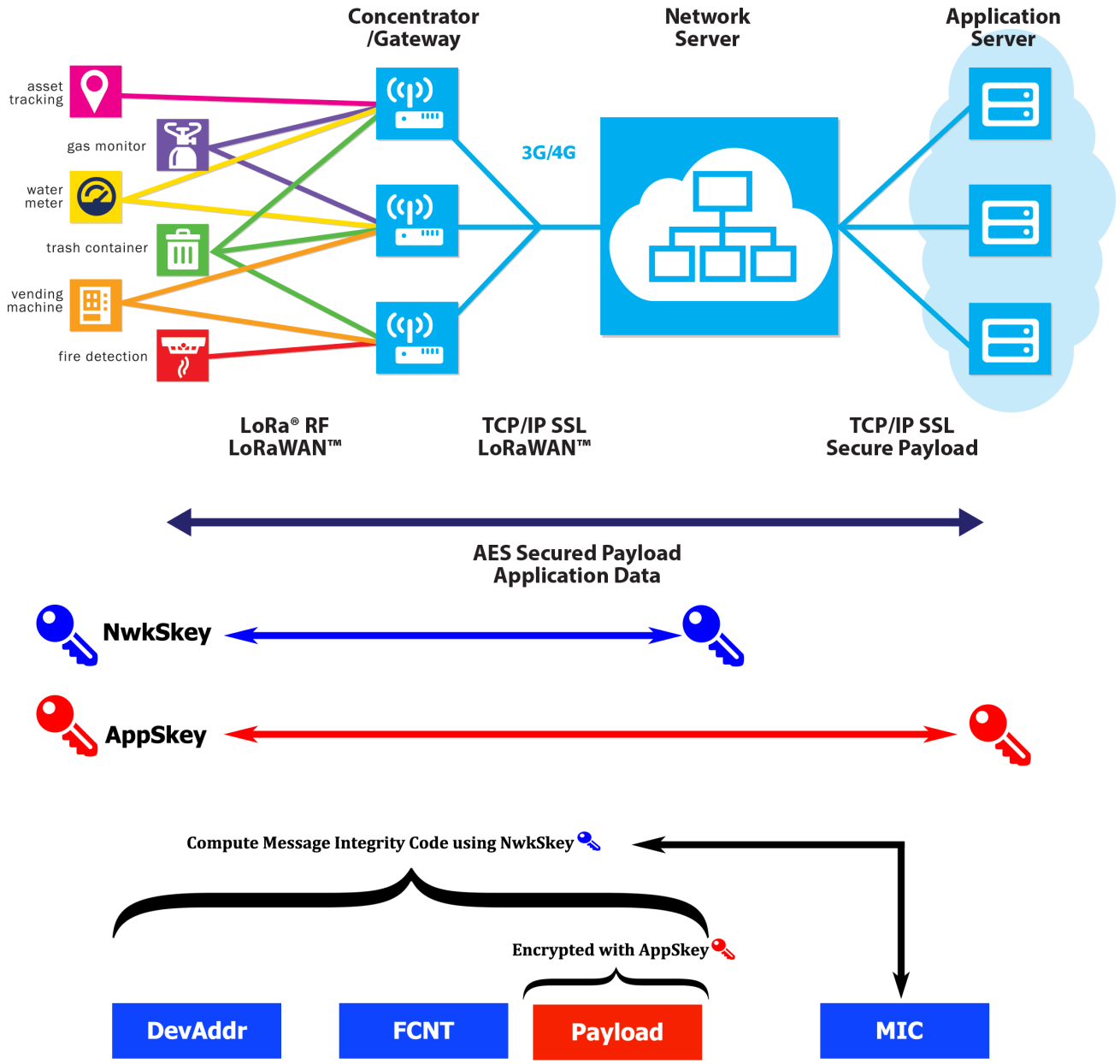


Figure 2: Security sceheme overview.[11]

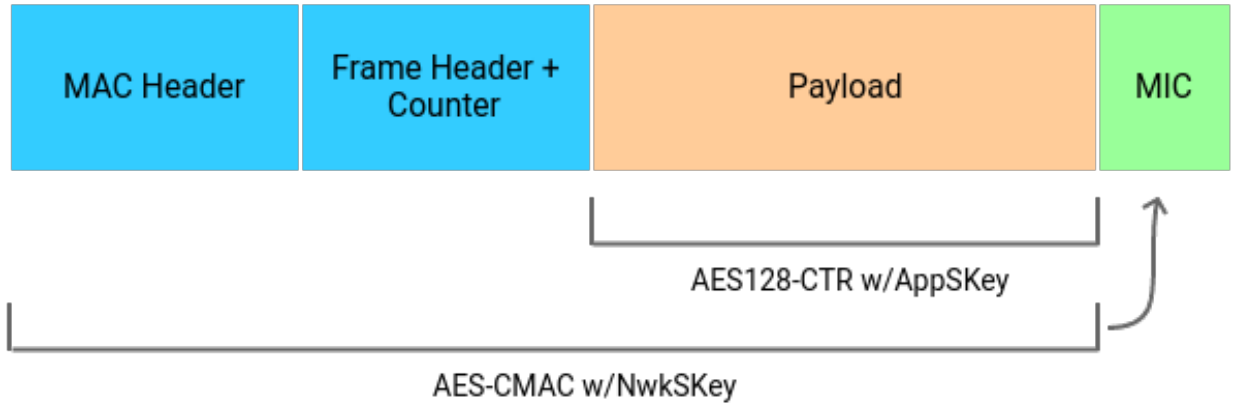


Figure 3: The LoRa frame.[12]

It is also worth noting that LoRaWAN supports three different classes of devices. Class A devices are currently most common. They can transmit data to the LoRaWAN gateway at any time, and then listen for a response from the gateway for a short window of 1-2 seconds. Class B devices are an extension of Class A, meaning that they will also listen for downlink messages at scheduled receive windows. Class C devices extend this even further by always listening for downlink messages from the gateway, unless they themselves are transmitting.

3. COMPONENTS

For this experiment, a set of different COTS devices were examined. These devices allow tracking of GPS positions and communication over LoRa and LoRaWAN. The devices examined allow for varying degree of control, further details on each device can be found in their respective sections.

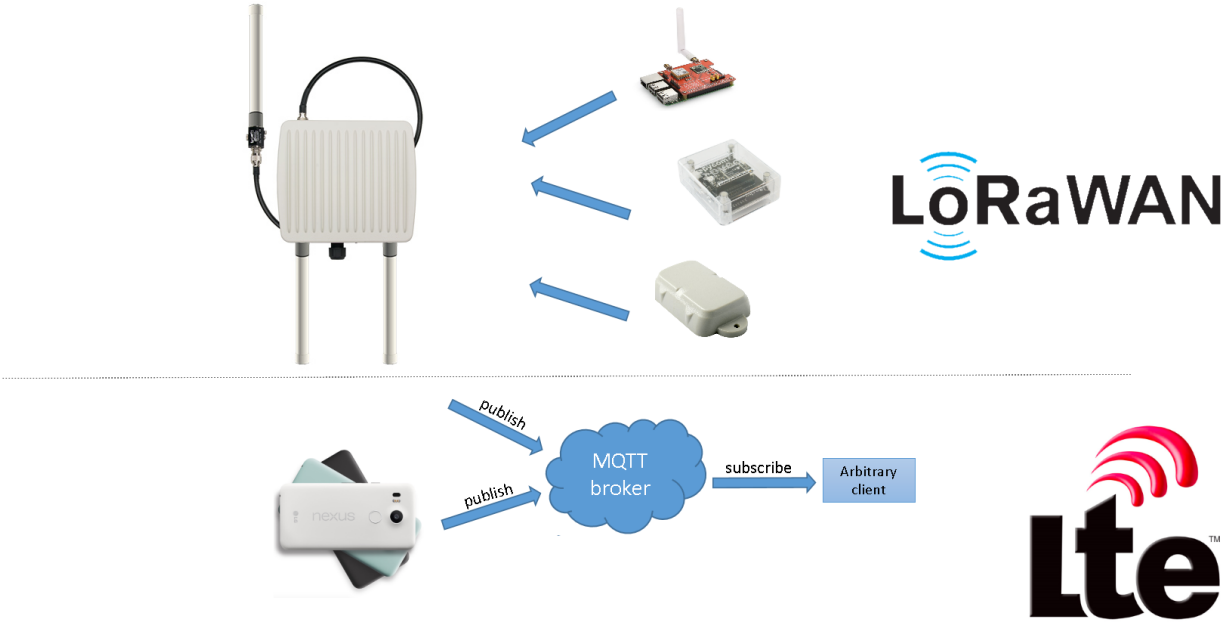


Figure 4: Information flow overview.

Figure 4 gives an overview of the information flow in our setup. Here, all LoRaWAN capable devices communicate with the LoRaWAN gateway, which also communicates over the cellular network. In our setup, we set the gateway to pass events to an online server. Comparing our deployment to the architecture in Figure 2, we have the gateway, the Network Server is realized through the MQTT broker, whereas the Application Server is represented by what we call Arbitrary client. Our smartphones (used for reference positions) reported directly to this same server. In Section 3.8 you can find an inventory list for this experiment. The remainder of this section elaborates further on the components of our experimental setup.

3.1 Raspberry Pi 3 with Dragino HATs

The Raspberry Pi is a low cost computer about the size of a credit card.[14] It is capable of running standard operating systems (commonly Linux distributions) and thus has access to a wide range of software and technology platforms.

Using the General Purpose Input/Output (GPIO) interface, an extension board, a so-called Hardware Attached on Top (HAT), can be attached to expand upon the abilities of the already versatile Raspberry Pi. The Dragino HAT[15] is such an extension board that allows the Raspberry to communicate with GPS satellites and LoRa devices.

For this experiment, the Raspberry Pi 3 Model B Rev 1.2 and Dragino LoRa/GPS HAT v1.4 were used.

3.1.1 Basic setup for Raspberry Pi

First we had to install a Linux image to a microSD card, as explained in.[16] We chose Raspbian Stretch Lite as it is a lightweight option, allowing the device to boot quickly.

After the image was written to the microSD card, we could boot the system. The next step was to edit some of the configuration files on the Raspberry. The options we changed were:

- connecting to the network (wireless or Ethernet)
- enabled Secure shell (SSH) to allow remote access
- enabled SPI (Serial Peripheral Interface) under interface options so the Dragino board is able to write GPS data to the Raspberry Pi
- updated keyboard layout to match a Nordic keyboard

Then, we fully updated the system and installed Python tools.

3.1.2 LoRaWAN setup for Raspberry Pi

The libraries have been updated since our previous work,[4] making the process of connecting and communicating using LoRaWAN easier than with the previous version. We used the Dragino v0.0.2 library[17] by Computenodes. This library is a fork of a more general library,[18] tuned for the Raspberry Pi + Dragino HAT combination.

We set up the authentication mode to be OTAA, and configured the AppKey, AppEUI and DevEUI, respectively. These values were then later entered in the setup of the gateway, thus allowing the devices to connect.

3.1.3 GPS setup for Raspberry Pi

To get the GPS working on the Dragino HAT we needed to install the gpsd service and a Python library[19] for that service. To allow the Python GPS library to access the GPS socket, we had to disable the gpsd service. We also had to to disable the getty service as we used the ttyS0 socket. Finally, we enabled the universal asynchronous receiver-transmitter (UART). With this in place, we were able to receive and monitor the GPS signal on the Raspberry Pi.

3.2 Pytrack and LoPy4

The Pytrack is a sensor board produced by Pycom.[20] Its sensory features include GPS, GLONASS and a 3-axis accelerometer. In addition to the Pytrack boards, an extension board called LoPy4 was used. It supports WiFi, LoRa, Bluetooth Low Energy (BLE) and Sigfox. WiFi was used when setting up and configuring the devices in the lab. LoRa was used to support LoRaWAN in our experiments.

The Pytrack uses a microSD card and runs a flavor of Python 3 called MicroPython,[21] optimized for microcontrollers. Unlike Raspberry Pi, Pytrack does not run a full-featured OS, but rather tries to execute the file main.py on the device. If this file is not found, it will try to execute boot.py. For our experiments, we developed our own main.py to replace the example one provided with the devices. We refer to these devices as "Pycom" in the remainder of the paper.

The setup for the Pycom devices was done following the official tutorial.[22] The firmware version was updated to pytrack0.0.8 and the libraries for LoRa and GPS tracking were updated to version 1.0.1.

3.3 Oyster GPS trackers

Oyster is a COTS product by Digital Matter for asset tracking.[23] It features both GPS and GLONASS support, has an accelerometer and is able to communicate over LoRaWAN. It is rated IP67, allowing for use in rough weather conditions. Use of ultra-low power modes allows for up to 5 years battery life, running on standard AA batteries.

Although the Oysters are configurable to allow different use cases, we used them as they were delivered from the vendor. The DevEUI, default AppKey and default AppEUI are printed on the back of the device. We registered these in the gateway's configuration, enabling the devices to communicate.

The Oysters send their data in a compact data format described here.[24,25] We adopted parts of this encoding scheme for the Pycom and Raspberry Pi devices, as it provided a compact way of sending the positional data.

3.4 LoRaWAN gateway

We obtained a Multitech Conduit IP67 LoRa gateway with LTE support.[26] This is a robust gateway, with Power over Ethernet (PoE) as the power source. The gateway was deployed outside in an elevated location (on the roof of FFI) for the duration of the experiments.

We used the gateway's built-in web administration tool to configure it. We installed a SIM card for Internet reachback, and added all our LoRa devices to the gateway as Class A devices. This mode fits well with the asset tracking application we are investigating, where our devices are only expected to provide data. The results from using the tool are written to a configuration file that is used by the gateway. If, for some reason, using the tool is not desirable, it is also possible to edit this file directly.

The gateway comes with a built-in Mosquitto MQTT broker. For security reasons, we chose not to expose the gateway over LTE on the Internet, but rather push data received over LoRaWAN to an external MQTT broker. MQTT is further described in Section 3.6.

3.5 OwnTracks

In addition to the LoRa devices, a set of smartphones were used to get reference points when tracking. The idea was that by collecting GPS data through a parallel and reliable channel (LTE network) we would have a representation of our assets' movements to compare that of our, in some areas, presumably less reliable LoRaWAN setup.

Data collection was mostly performed in urban or suburban environments, where smartphones are known to be reliable and accurate. For this experiment, three Google Nexus 5X Android smartphones[27] were factory reset. Then, the open source tracking app, OwnTracks[28] was installed. The app was configured to publish positional and temporal data to the same MQTT broker as the LoRa devices, under the `owntracks/` topic. The payload the app delivers is in JavaScript Object Notation (JSON) format and therefore easy to parse.

3.6 MQTT

Event-driven message exchange, or publish/subscribe as it is often called, is a message exchange pattern in which entities that have information they want to share (i.e., producers or publishers) can publish this information. Information consumers (i.e., consumers or subscribers) can subscribe to specific types of information they want to receive. When information is published that matches the subscribed interests, it is sent to the subscriber(s). The distribution of information is often performed by a message broker.

MQTT is an ISO standard (ISO/IEC PRF 20922) which is built on the TCP/IP protocol. There are different versions of the standard. We're using the MQTT v3.1.1 standard, which is mature and well supported these days. For details on the workings of MQTT and the topic structure it adheres to, see the discussion in.[4]

In addition to having a single broker, there is an approach to achieving multi broker setups that uses the standard API to build a so-called MQTT bridge. This is the approach we used in this experiment, in that we bridged the MQTT broker in our LoRaWAN gateway to an MQTT broker installed in the cloud. By doing this we did not have to expose the gateway on the Internet, but could connect our external MQTT clients, visualization software and log analysis tools to the open MQTT broker in the cloud instead, where LoRa data from the gateway was being mirrored through the MQTT bridge. Here, all LoRa capable devices reported using the `lora/` topic, while the OwnTracks devices reported on the `owntracks/` topic.

3.7 GNU Radio and Software Defined Radios

Intercepting radio traffic is typically done using Software Defined Radios (SDRs). Investigating the feasibility of intercepting and tampering with LoRaWAN traffic is useful, in order to determine the capabilities necessary for an adversary. Through traffic analysis he may be able to gain information on the data being sent, the location of the sensor and the assets being tracked, the number of sensors and assets in an area, and the location of the gateway. The process of both intercepting and transmitting LoRa packets using SDRs have been studied earlier, though in the US frequency band.[29] For this research, we wanted to replicate our previous studies under European conditions, and we wanted to investigate methods for improving the interception ranges.

In order to intercept LoRa traffic, USRP B210 SDRs were used along with the GNU Radio software tool. GNU Radio is a popular choice for analyzing and decoding traffic that is either captured or transmitted using an SDR. There are a couple of modules available for decoding LoRa frames, where we chose to use the gr-lora module created by Robyns et al.[30]

By using this setup, it is possible to capture LoRa traffic for a single frequency channel. As LoRaWAN by default hops between eight different channels, it would be necessary to have one SDR listening to each of the frequencies in order to capture all packets.

3.8 Device summary

Table 1 gives an inventory list of the hardware and versions used in the experiment.

4. RANGE TESTS AND ASSET TRACKING RESULTS AND ANALYSIS

We performed several field trials with the above mentioned setup. Data collection was done by driving around with multiple devices in each car. In addition to the three types of LoRa capable devices (Raspberry Pi, Pycom and Oyster), there was also a smartphone with the OwnTracks app running as a reference point. We had created logging software (running on the `Arbitrary client` shown in Figure 4), which was configured to log and listen to all traffic coming from the MQTT broker. In Table 2 you can see an overview of the different datasets collected related to this paper, which devices were used, and how they were allocated. The `bruker__` and `lora_` devices in the data set refer to OwnTracks and Dragino + Raspberry Pi devices, respectively. See Figure 5 for device deployment examples.

Unfortunately, we encountered several technical issues during the experiment. These challenges took up a lot of time and forced us to downscale some aspects of the testing. Hence, we include only results from a subset of our experiments in this paper, as the prior tests were inconclusive. That said, data from the other experiment

Table 1: Inventory list for the experiment.

Sets	Name	Version
3x	**Raspberry Pi 3** Dragino LoRa/GPS HAT Raspbian Stretch Lite Python MicroSD card	**model B rev. 1.2** 1.4 2019-04-08 release 3.5 16GB
3x	**Pytrack** LoPy4 MicroPython MicroSD card	**0.0.8** v1.18.2.r1 v1.8.6-849-e0fb68e 16GB
3x	**Google Nexus 5X** Android OwnTracks	 8.1.0 (sec. updates to 2018/12/5) 2.1.2 (21201)
4x	**Oyster GPS tracker**	**LoRaWAN 868MHz**
1x	**Multitech gateway** MultiConnect Conduit	**MTCDTIP-LEU1-266A** Firmware 1.6.4
1x	**GNU Radio** gr-lora USRP B210	**3.7.11** 0.7

Table 2: Dataset overview.

Run & Date	Devices	Delegation
Drive 2 2019/07/15	2x OwnTracks 3x Dragino 2x Oyster	**Car A** - Bruker49 - Lora2 - Lora1 - Oyster2 **Car B** - Bruker50 - LoRa3 - Oyster3
Drive 3 2019/07/19	1x OwnTracks 3x Dragino	**Car A** - Bruker49 - Lora1 - Lora2 - Lora3

(a) LoRaWAN devices deployed for Drive 1 (one of two vehicles shown).

(b) All three Raspberry Pi 3 with Dragino HATs deployed in one vehicle for Drive 3.

Figure 5: Deploying LoRaWAN devices for tests.

runs can be found in our technical report[31] along with the scripts used to disseminate information, gather and analyze the data.

We first encountered problems with the Pycom devices during the first test. While connecting to the gateway and transmitting during the development of the software, everything seemed fine. However when the tests were performed, they lost contact. After debugging and still not finding an error in the code, a decision to revert to an old version of the GPS library was made. However the devices continued to display the same behavior. They worked fine when testing locally (close to the gateway), but lost connection during the test drives. The exact source of this error is still not known to us, though we anticipate that it is either caused by limitations in the Pycom hardware, or possibly related to a bug in either the firmware or libraries we were using.

As for the Oyster devices, we did not have the configuration hardware to change the rate of transmission. The default rate was too sparse for us to collect data that would be useful at the given timescale. Thus, we shifted focus to the Dragino devices, as they were the devices giving the most consistent and reliable data.

Regardless of technical problems we did manage to collect some data that gives insight into the practical application of these devices. In Figure 6 you can see an example of the data collected. The purple line is the path of the *bruker49* OwnTracks device and the yellow line is *lora1*, a Raspberry Pi. In Subfigure (a) the positional markers of the *lora1* device are shown. Here, places where the yellow path occurs as long, straight lines without markers, indicates a loss of reporting in that area. The purple line should be considered actual positions, since we had cell phone coverage for all our tests. So, we consider places where the yellow path does not follow the purple reference line to be where the *lora1* device was not able to reach the gateway (in places where the yellow line follows the purple line it will be hidden by the purple line, but markers show the *lora1* positions). Subfigure (b) shows a plot of the distance to the gateway at the different timestamps. Both the colors of the paths and the number markers correspond to each other between (a) and (b).

The number markers highlight some interesting points in the map and time-distance graph, and serve to give context about terrain and thus help us to better understand the graph. For example the stretch between point 5 and 6 is at a lower elevation from where the gateway is positioned, and therefore non-LoS (Line of Sight). Similarly the path along 2-3-4 is "hidden" behind a hill on the east side of the gateway location. At point 6 we get LoS going up the hill along a highway into the forested area, but lose it again on the other side. This behavior indicates that, in this environment, it is not the strength of the signal that is the limiting factor for range, but the terrain and ability to have LoS to the gateway.

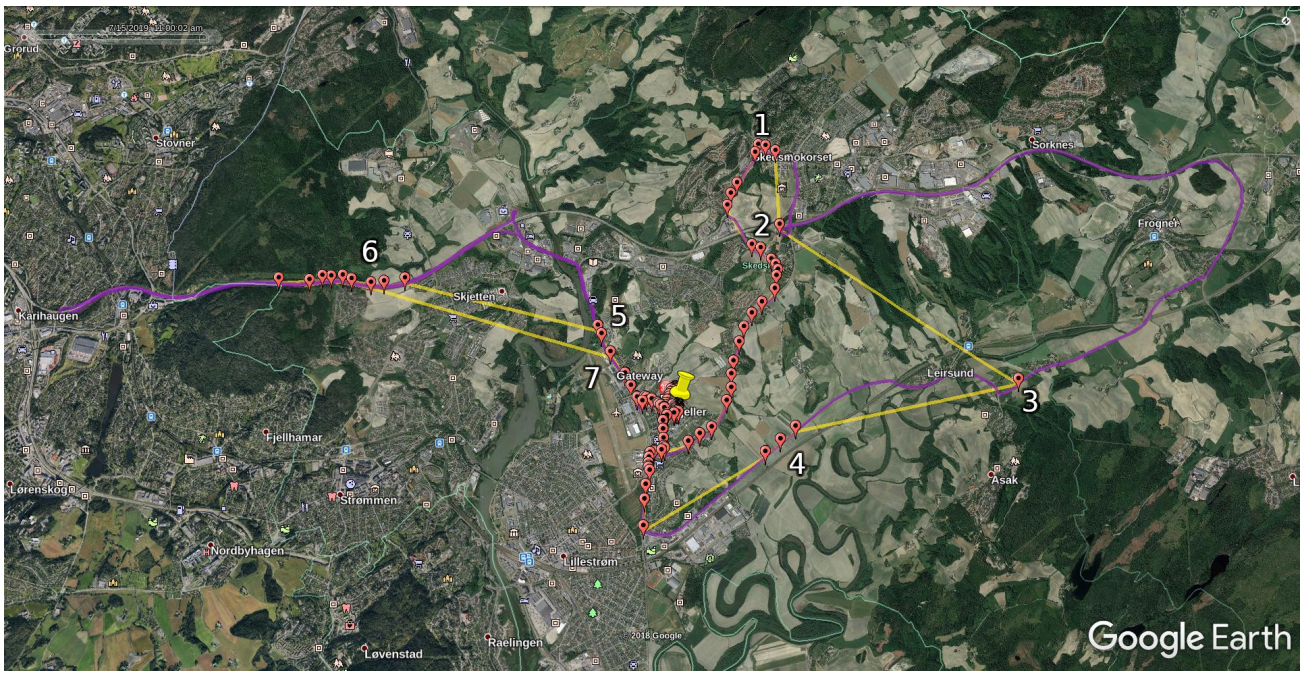
The environment where the tests were conducted are a combination of rural and semi-urban and mostly hilly terrain. In terms of raw distance, point 6 is about 4.7 km away from the gateway (other devices from the same test reached all the way to Fjerdingby, about 6 km away). This range seems to be in-line with what is to be expected, as stated in Chauhan and Lee:[32]

> "On average, in an urban environment with an outdoor gateway, you can expect up to 2- to 3-km-wide coverage, while in the rural areas it can reach beyond 5 to 7 km. In some cases, extremely long range is also attainable."

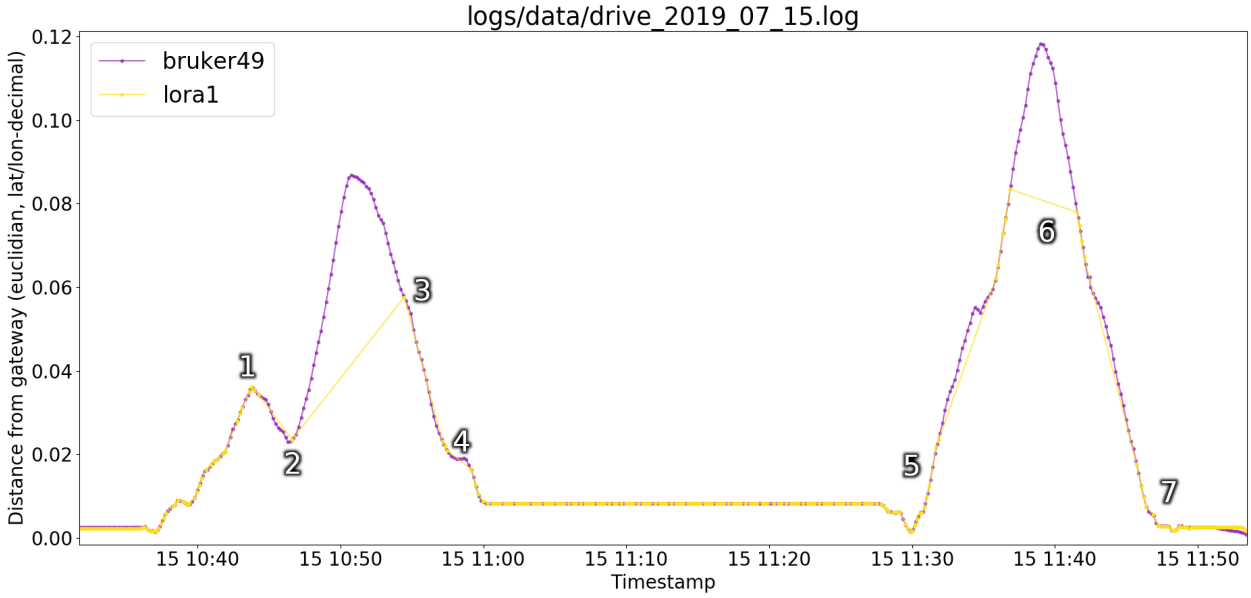
Note that the OwnTracks reference position (purple) is often hidden by markers. This is to be expected, as the LoRa reports overlay the reference reports in our graphs. Hence, it is easy to spot when the purple line is actually visible, which indicates that LoRa data was not gathered in that particular area. Drive 3 was conducted using the Dragino HATs yet again, and results from this experiment run supports the findings from Drive 2 presented and discussed above.

5. INTERCEPTING LORA

The security of LoRa and LoRaWAN has been described and analyzed by several other researchers, where some of the challenges pointed out are proper key management, as well as jamming.[10,33] Our focus is rather on the possibility of intercepting LoRa traffic and improving the results from our previous experiments.[29] As our previous setup consisted of using a HackRF SDR with a 900 MHz omnidirectional antenna as a capturing device, we wanted to see how the more powerful USRP B210 would compare in combination with directional antennas.



(a) Map showing the route of the devices, colored paths correspond to the labels below. Only the position markers for the Lora1 device and the gateway(pin) are enabled here.



(b) The number labels correspond to the position of the same number in the map.

Figure 6: Map showing route of a subset of the tracked devices and corresponding distance-time graph for Drive 2.

Our setup for intercepting LoRa was as follows: a Pycom sensor was configured to transmit a message every 5 seconds on 868 MHz using SF7 and BW125KHz, while a laptop running the GNU Radio script shown in Figure 7 was used to intercept the messages. We performed experiments using the three different antennas shown in Figure 8: an omnipolar antenna, a log periodic antenna, and a Yagi antenna. We found the omnipolar antenna to have a maximum range of about 70 meters, while the log periodic antenna had a range of ca 480 meters. We got the best results with the Yagi antenna, which worked at a range of 800 meters. Note that we were not able to achieve LoS beyond these 800 meters, but suspect the Yagi antenna to work at even greater ranges, provided LoS.

Unlike the experiments with range discussed previously, which were conducted using vehicles, this part of the experiments were conducted on foot. Hence, it was not possible to revisit all areas that were previously used, as e.g., walking on the freeway is both dangerous and prohibited. Also note that both the log periodic antenna and the Yagi antenna are directional, and achieve best results when pointing directly at the transmitter.

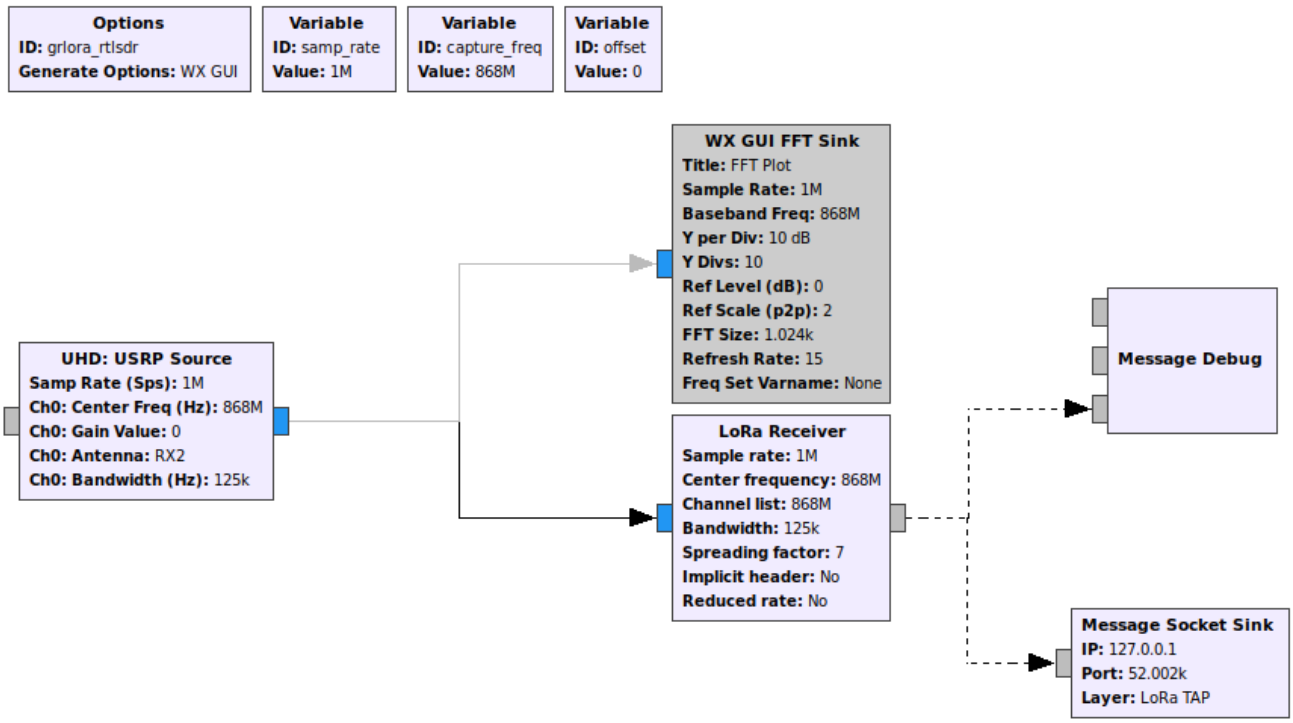


Figure 7: The GNU Radio script for intercepting LoRa.

6. RELATED WORK

In,[4] we investigated LoRa as a protocol for blue force tracking. Here, we used Raspberry Pi and Dragino HATs to disseminate GPS positions. Different spread factors, code rates and bandwidths were investigated. Experimenting with LoRaWAN was identified as work to be done. Hence, that paper can be considered the precursor to the work we presented here.

Jalain et al.[34] investigated LoRaWAN as a protocol in the tactical domain. These experiments used Pycom devices and a Multitech gateway (similar to parts of the equipment we used, only their experiments were conducted in the USA and thus the devices operated in the 915 MHz band). The work had a twofold contribution; it showed the feasibility of using LoRaWAN at the tactical edge, as well as integrating information from the IoT devices with military C2 systems. In this experiment, ranges up to 6.1 miles (approximately 9.8 km) were achieved.

Michaelis et al.[3] evaluated LoRaWAN in an urban environment, for the tracking of vehicles. Like the previous work, Pycom units were used in the 915 MHz ISM band. In these experiments, messages could be received as

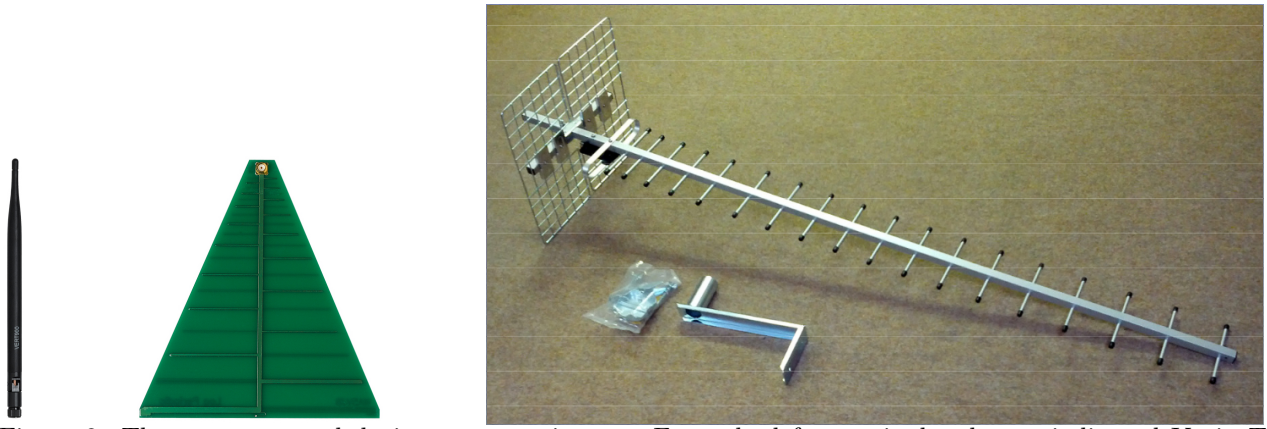


Figure 8: The antennas used during out experiments. From the left: omnipolar, log periodic and Yagi. The images are not to scale.

far as 5.5 km from the gateway. In the case where buildings obstructed the line of sight, packet loss increased and the effective range was shorter, around 2.5 km.

Aras et al.[33] explored various security vulnerabilities in the LoRa stack. They found, among other things, possibilities for both jamming and replaying LoRa messages.

Our previous research[29] looked at the possibilities for both intercepting and transmitting LoRa traffic using SDRs. It concluded with both being possible using cheap COTS hardware in combination with open source software, although the ranges achieved were not comparable to those possible with LoRa sensors.

7. CONCLUSION

In these experiments, we had three goals. Our first goal was to evaluate COTS IoT LoRaWAN products, and as explained in this paper we were able to obtain, configure, and use a variety of such products. Except for the purchased hardware, all software used was either free (like OwnTracks) or free and open source (the remainder of software used). This is in line with current IoT trends, where very affordable yet capable devices can be obtained and used for various purposes. We did experience, however, a difference in how well the devices performed (at least the way we attempted to use them). We got good, consistent results using Raspberry Pi 3 with LoRaWAN HATs from Dragino, whereas we got intermittent problems and somewhat inconclusive results from our use of the Pycom modules. This does not necessarily imply that the Dragino HATs are any better than the Pycom modules, but it does show that for the specific combination of hardware and software that we chose, the Pycom modules did not exhibit the same reliability as the Raspberry Pi setup.

Our second goal was to determine how LoRaWAN fares as the underlying protocol for asset tracking. To support this application, we configured our devices as so-called Class A, meaning that they would perform only one-way communication to the gateway (sending their tracking information). This functioned as expected, so when they were in range of the gateway the devices were able to connect and send their information. Further, the bridging between the LoRaWAN gateway and MQTT broker on the Internet over the cellular network also performed admirably. Here, we were able to get both LoRaWAN events and OwnTracks events in the same server (albeit using different MQTT topics), so that the subscription and analysis software we developed could easily obtain data in near real time from this server.

Our third goal was to investigate the feasibility of intercepting LoRa traffic using SDR radios. This was done using a USRP B210 and GNU Radio. We found that directional antennas improved the range when intercepting, though they are reliant on line of sight. By using this setup in an asset tracking scenario, an adversary would be able to determine the location and number of assets and possibly pinpoint the exact vehicles in an urban area. Although the LoRaWAN traffic is encrypted, the security relies on correct use and handling of encryption keys and credentials. Should one of the sensors fall in the hands of adversaries, they may be able to recover both AppSKey and NwkSKey.

In conclusion, we can say that LoRaWAN can be used for asset tracking purposes, and that the ranges we experienced (several kilometers in actual use) are in line with what vendors and other experiments have reported. So, it does seem that the European version of LoRaWAN that we used (at 868 MHz) does indeed perform similarly to what has been reported from experiments using the USA version, though it uses another frequency.

Finally, with respect to LoRaWAN and security, challenges already identified in literature are proper key management, as well as jamming. In our work we have investigated intercepting LoRaWAN traffic, and found that it was feasible using affordable hardware and open source software. The limitations are in the antennas used. We found that with an omnipolar antenna the range to eavesdrop was up to 70 meters. Switching to a log periodic antenna increased the range to 480 meters. Finally, using a Yagi antenna the range increased to 800 meters (limited not by the antenna, but the fact that we were not able to achieve LoS beyond 800 meters).

8. FURTHER WORK

There are several areas that are left open for further research. The most obvious is to address the remaining technical issues, in order to further test the devices. It would also be interesting to study how larger infrastructures affect the capabilities of the technology. Especially looking into using multiple gateways and relaying of messages. A third area of interest would be to look into effective range in woodlands and mountainous areas.

There are also interesting areas for research related to LoRa and LoRaWAN security. We have shown how it is possible to intercept LoRa packets even over long distances under the right conditions. As directional antennas highly depend on the location of the sensors, it would be interesting to examine the possibilities of triangulating devices using multiple SDRs. It would also be interesting to analyze the metadata and packet format of encrypted LoRaWAN traffic — both during initialization and during transmission. Lastly, it would be interesting to analyze the data sent from the LoRaWAN gateway.

Acknowledgment

Thanks to Ketil Lund and Trude H. Bloebaum for support during the range experiments, and to Anders Ødegård for support and insight during the interception experiments. Last but not least, thanks to Philip Ø. Puente for his efforts coding and visualizing results in the LoRaWAN range tests.

REFERENCES

[1] N. Suri, M. Tortonesi, J. Michaelis, P. Budulas, G. Benincasa, S. Russell, C. Stefanelli, and R. Winkler, "Analyzing the applicability of Internet of Things to the battlefield environment," 2016 International Conference on Military Communications and Information Systems (ICMCIS), 23-24 May, Brussels, Belgium, 2016.

[2] IoT Special Interest Group, "Technology Strategy Board," 2013.

[3] J. Michaelis, A. Morelli, A. Raglin, D. James, and N. Suri, "Leveraging LoRaWAN to Support IoBT in Urban Environments," IEEE World Forum on Internet of Things (WF-IoT) 2019, 16 April 2019, Special session on military applications of IoT, Limerick, Ireland.

[4] F. T. Johnsen, T. H. Bloebaum, and P. Ø. Puente, "Towards friendly force tracking with MQTT over LoRa," 24th International Command and Control Research and Technology Symposium (ICCRTS), Laurel, MA, USA, 2019.

[5] "LoRa," Dec. 2019, page Version ID: 929212123. [Online]. Available: https://en.wikipedia.org/w/index.php?title=LoRa&oldid=929212123

[6] IoT ONE, "Chirp spread spectrum (css)," https://www.iotone.com/term/chirp-spread-spectrum-css/t110, 2019.

[7] D. Purves, "Low Power Wireless Sensor Network Technologies and Standards for the Internet of Things," 2016. [Online]. Available: https://www.slideshare.net/DuncanPurves/low-power-wireless-sensor-network-technologies-and-standards-for-the-internet-of-things

[8] The Things Network, "LoRaWAN Frequencies Overview," https://www.thethingsnetwork.org/docs/lorawan/frequency-plans.html#eu863-870, accessed 2019-08-30, 2019.

[9] LoRa Alliance Technical committee, "LoRaWAN 1.0.2 Regional Parameters," Feb. 2017. [Online]. Available: https://lora-alliance.org/sites/default/files/2018-05/lorawan_regional_parameters_v1.0.2_final_1944_1.pdf
[10] R. Miller, "LoRa Security - Building a Secure LoRa Solution," 2016. [Online]. Available: https://labs.f-secure.com/assets/BlogFiles/mwri-LoRa-security-guide-1.2-2016-03-22.pdf
[11] resiot.io, "What is lorawan™? how it works? technical specifications and applications," https://www.resiot.io/en/what-is-lorawan/, 2019.
[12] "Ensuring Device & Radio Security in LoRaWAN | DigiKey." [Online]. Available: https://www.digikey.com/en/articles/techzone/2017/aug/ensuring-device-radio-security-lorawan
[13] Gemalto, Actility and Semtech, "LoRaWAN Security - Full End-To-End Encryption for IoT Application Providers," Feb. 2017. [Online]. Available: https://lora-alliance.org/sites/default/files/2019-05/lorawan_security_whitepaper.pdf
[14] The Raspberry Pi Foundation, "Raspberry Pi Model 3 B," Retrived July 22nd, 2019, from https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.
[15] Dragino, "LoRa GPS HAT for Raspberry Pi," Retrived July 22nd, 2019, from http://www.dragino.com/products/module/item/106-lora-gps-hat.html.
[16] Raspberry Foundation, "Installing operating system images," Retrived July 19th, 2019, from https://www.raspberrypi.org/documentation/installation/installing-images/.
[17] Computenodes, "LoRaWAN implementation in python for Dragino hat," Retrived July 19th, 2019, from https://github.com/computenodes/dragino.
[18] Jeroennijhof, "LoRaWAN implementation in python," Retrived July 19th, 2019, from https://github.com/jeroennijhof/LoRaWAN.
[19] Wadda, "gps3 0.33.3," Retrived July 29th, 2019, from https://pypi.org/project/gps3/.
[20] Pycom, "Pycom - Go invent," Retrived July 30th, 2019, from https://pycom.io/.
[21] Damien George, "Micropython," Retrived July 30th, 2019, from https://micropython.org/.
[22] Pycom, "Documentation and setup," Retrived July 30th, 2019, from https://docs.pycom.io/.
[23] Digital Matter, "Oyster — Compact IP67 Battery GPS Asset Tracking on LoRaWAN Networks," Retrived July 19th, 2019, from https://www.digitalmatter.com/Devices/Lorawan-GPS-Trackers/Oyster-Lorawan.
[24] ——, "Oyster — LoRaWAN Integration," Retrived July 19th, 2019, from https://digmat.freshdesk.com/support/solutions/articles/16000066131-oyster-lorawan-integration.
[25] ——, "Decoding the Oyster LoRaWAN Payload ," Retrived July 23rd, 2019, from https://support.digitalmatter.com/support/solutions/articles/16000069424-decoding-the-oyster-lorawan-payload.
[26] Multitech, "MTCDTIP-LEU1 Home Page," https://www.multitech.com/models/94557480LF, 2019.
[27] Google, "Nexus tech specs," Retrived July 22nd, 2019, from https://support.google.com/nexus/answer/6102470?hl=en.
[28] OwnTracks, "OwnTracks — Your location companion," Retrived July 22nd, 2019, from https://owntracks.org/.
[29] T. Søndrol, B. Jalaian, and N. Suri, "Investigating LoRa for the Internet of Battlefield Things: A Cyber Perspective," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, Oct. 2018, pp. 749–756, iSSN: 2155-7578.
[30] P. Robyns, P. Quax, W. Lamotte, and W. Thenaers, "rpp0/gr-lora," Dec. 2019. [Online]. Available: https://github.com/rpp0/gr-lora
[31] P. Ø. Puente and F. T. Johnsen, "Asset tracking experiments with lorawan," FFI-Eksternnotat 19/01760, 2019.
[32] Rishabh Chauhan, Keith Lee, "Electronic Design — 11 Myths About LoRaWAN," January 25th 2018, Retrived July 24rd, 2019, from https://www.electronicdesign.com/industrial-automation/11-myths-about-lorawan.
[33] 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)*, Jun. 2017, pp. 1–6, iSSN: null.
[34] B. Jalaian, T. Gregory, N. Suri, S. Russell, L. Sadler, and M. Lee, "Evaluating LoRaWAN-based IoT devices for the tactical military environment," 4th IEEE World Forum on Internet of Things (WF-IoT) 2018, Singapore, February 5-8, 2018, 2018.