TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Karl Hendrik Leppmets

SECURE DATABASE SYNCHRONIZATION OVER EXTREMELY LOW BANDWIDTH DATALINKS IN MILITARY ENVIRONMENT

Bachelor's thesis

Supervisor: Kaido Kikkas
PhD

Tallinn 2020

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Karl Hendrik Leppmets

TURVALINE ANDMEBAASIDE SÜKRONISEERIMINE ÜLE VÄGA VÄIKESE KIIRUSEGA ANDMESIDEÜHENDUSTE MILITAARKESKKONNAS

Bakalaurusetöö

Juhendaja: Kaido Kikkas
PhD

Tallinn 2020

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Karl Hendrik Leppmets

18.05.2020

Abstract

The aim of the current thesis is to design theoretical proof-of-concept of low bandwidth-based database synchronization system that could be used in the Estonian Defence League. The work will give an overview of the key components that are planned to be used in the final product. Additionally, system design and logic will be discussed. As Estonia's most probable adversary is advancing their Electronic Warfare capabilities then also Estonia has to keep up with modern solutions to counter those issues.

This thesis is written in English and is 39 pages long, including 5 chapters, 7 figures and 5 tables.

Annotatsioon

Käesoleva diplomitöö eesmärk on luua teoreetiline kontseptsioonitõestus madalal andmesidekiirusel toimivast andmebaaside sünkroniseerimise süsteemist Eesti Kaitseliidu jaoks. Töö annab ülevaate olulistest komponentidest, mida on plaanis kasutada lõpplahenduse jaoks. Lisaks on vaatluse all ka üldine süsteemi disain ja loogika. Kuna Eesti kõige potentsiaalsem sõjaline vastane arendab väga aktiivselt oma elektroonilise sõjapidamise võimekust, siis peab ka Eesti astuma samme, et saada vajadusel vastu sellest tulenevatele probleemidele

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 39 leheküljel, 5 peatükki, 7 joonist, 5 tabelit.

List of abbreviations and terms

EW	Electronic warfare
LoRa	Low-power wide-area network technology
SIGINT	Signals intelligence
OSINT	Open-source intelligence
AES	Advanced Encryption Standard
DES	Data Encryption Standard
RSA	RSA algorithm (Rivest-Shamir-Adleman)
CPU	Central processing unit
IBM	International Business Machines Corporation
NSA	National Security Agency (USA)
GPS	Global Positioning System
IP	Internet Protocol (address)
UDP	User Datagram Protocol

Table of contents

List of figures

List of tables

Introduction

This thesis is about designing and starting the development of a prototype for a secure database synchronization system using LoRa technology for Estonian Defence League. The reason for this is the limitations of current systems for specific needs of Defence League and the price and unavailability of similar new military-grade systems.

The work will give a brief overview of existing communication methods, their limitations, and possible dangers posed by potential adversaries. Additionally, tools and methods used in military forces to detect or interfere with communication methods will be explored.

By the end of this thesis, there will be a design of proof of concept of LoRa based database synchronization and communication tool with analysis if it actually could be suitable for Estonian Defence League's needs.

The potential conflict scenarios in this thesis between nations are created by the author and do not represent the actual situation between mentioned nations.

1 Description of the problem and formulation of the assignment

1.1 General Overview

During this century, the amount of information systems in military environments has grown just as fast in every other civilian business sector. When comparing the military world to civilian, then there are some fundamental differences. If in civilian world business IT systems and infrastructure are targeted mostly for monetary gain, then in the military world, the stakes are much higher. This requires gear and software that is considered military-grade, which in turn rises the zero count on the price by one or even two zeros.

Additionally, because every nation wants to protect their freedom and sovereignty, then usually products are not sold to non-allied nations nor is the research on such topics shared even with allies. This means that smaller nations with smaller defence budgets cannot afford expensive tools and hardware and motivates them to do the research on amongst their own ranks.

1.2 Background information

1.2.1 Current communication methods

Every military operation, no matter the scale, needs communication methods to relay information between different units. That can be reconnaissance findings about opposing forces locations and movements, allied forces needing supplies, information about friendly minefields.

Nowadays, the primary method of communication is radio-based communications. Biggest drawback of that is that adversary can also use radios or more sophisticated machinery to detect and listen in on allied force radio communications. To avoid that several countermeasures are in place:

1. Encryption
2. Frequency hopping
3. Directional antennas

Each method has its advantages and drawbacks. When using encryption options, then radio chatter is encrypted with pre-shared keys using different encryption methods. E.g. AES 128 and AES 256 encryption. If the keys are changed every now and then, then it makes it nearly impossible for an adversary to decrypt it and if opposing force captures friendly radio device it will be rendered useless without the new key. Encryption is known to reduce the distance over what radio communication sessions can be held.

Frequency hopping can be used alone or with encryption. Frequency hopping makes it even harder for the opponent to record and listen in on allied communications as frequencies are changed thousands of times per second. Different hopping algorithms and seeds can be used.

Directional antennas emit radio waves only in the intended direction, which makes it harder for opposing force to hear this from the dark side of the antenna. Biggest drawback about directional antennas is that every unit that accomplice’s location needs to be known in order to direct the antenna at them. Also, some command centres antennas will still be directed towards front lines which makes it even easier for the adversary to locate command points [1].

1.2.2 Electronic Warfare

During present times most of worlds militaries possess EW units. EW falls under SIGINT category, and their main task is to detect contents of an opponent's communications and directions or locations and to interfere/jam detected frequencies. EW units and their capabilities are usually one of the most secretive topics, and this is why there is not much information publicly available about the hardware and methods that are used. However, the overall logic is the same.

Either person or automated system is monitoring the selection of the band and when received power change is detected on some frequency, then captured data will be demodulated and turned into listenable audio. Additionally, directions can be measured.

By using several physical locations and their directions, an exact location can be calculated. By relaying this information upwards in chain of command, an indirect fire or air force intervention can be called upon the discovered location. Alternatively, an additional reconnaissance unit will be dispatched to the location to find out more about possible opponent's forces.

Discovered frequencies can be jammed, and this renders this frequency most probably unusable for opponents' forces. Additionally, fake commands can be broadcasted in order to get more information about units' readiness and supplies or just to direct them right into an ambush.

Figure 1 displays one example of possible EW antennas, gear, and machinery that is used for that EW purposes.

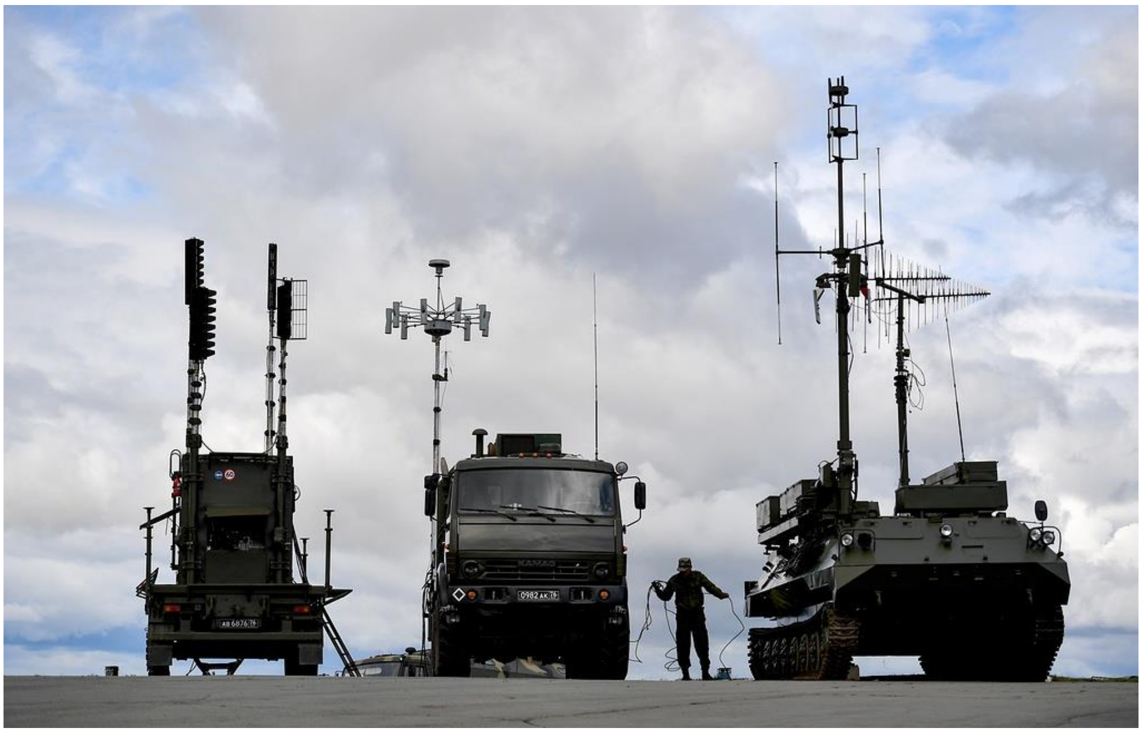
Figure 1. Russian EW unit systems in Syria [2]

Russia is known to be very active and professional when it comes to EW [3]. This can be concerning to Estonian defence forces in case of an armed conflict between the nations as staying hidden is one of the critical elements. Estonian Defence League units have started looking into different methods to mitigate possible risks and because authors involvement with Estonian Defence League, the author decided to design proof-of-concept using LoRa technology.

1.3 Description of the problem and goals

Because of LoRa’s limitations, casual audio-based communications cannot be used. Instead, a simple chat app will be used. Goal is to design proof-of-concept for a system capable of synchronizing message database over LoRa radio links, and to find out if this solution even could be feasible for future military use.

1.4 Limitations

Initially, this thesis was meant to include testing of LoRa devices for more accurate statements. Furthermore, an actual proof of concept was planned to be finished by the end of the thesis. All this was not possible due to lockdowns in different parts of the Earth due to COVID-19, and due to that, some companies were not able to ship devices needed. Author ordered devices shortly after submitting Draft of Thesis. By the time of submitting this thesis, some devices needed for the practical part of the thesis still have not reached the author. Author received some of the devices but too late and not in the required quantities to use them in order to finish the thesis as planned initially.

2 Methods and tools

2.1 Overview of methods

The main part will be split into three separate sections:

1) Overview of the tools and requirements of the system

2) Designing of proof-of-concept

3) Analysis of the suitability of the proof-of-concept in the military environment

Overview of the tools and requirements of the system will explain the inner workings of named tools and will give a more detailed overview of what is expected from the system. Design of the proof of concept will give an overview of different components in the system and analysis why one was chosen over the others. The final analysis will help to prove or disprove certain assumptions that were made during the design of proof-of-concept and to determine if this theoretical proof-of-concept could actually be turned into a real tool used in a military environment.

Using LoRa for this project was dictated by Estonian Defence league as author voluntarily picked up this project idea from there. LoRa was decided to be used by Estonian Defence League because of low powered transitions. This makes it really hard for opponent’s EW units to detect LoRa broadcasts in casual warfare settings. LoRa can be detected if EW units are as close as other LoRa nodes in the network.

2.2 Overview of tools

Tools that will be used for proof-of-concept or tools that are expected to be used for the final version:

1) C# – a programming language that ideally will be used for the final version.

2) Linux operating system (Ubuntu) – will be considered as a base operating system for the machines running the tool.

3) Rider – a software development tool meant for C# and .NET development [4].

4) Raspberry pi 4 – a low-cost small computer that is expected to run all services that are part of the projects network infrastructure [5].

5) RAK7224 LPWAN Developer Gateway – LoRa gateway developed by Shenzhen RAKwireless Technology Co., Ltd. [6]

6) Turtleboard – LoRa device developed by HelTec Automation [7].

7) Visual Paradigm Online Diagrams – An online software that is used to create diagrams used in this thesis and later in the documentation [8].

3 Components analysis

3.1 Overview of LoRa

LoRa (short for long range) is a spread spectrum modulation technique derived from chirp spread spectrum (CSS) technology. Semtech’s LoRa devices and wireless radio frequency technology is a long-range, low power wireless platform that has become the de-facto technology for Internet of Things (IoT) networks worldwide. LoRa devices and the open LoRaWAN® protocol enable smart IoT applications that solve some of the biggest challenges facing our planet: energy management, natural resource reduction, pollution control, infrastructure efficiency, disaster prevention, and more. Semtech’s LoRa devices and the LoRaWAN protocol have amassed several hundred known uses cases for smart cities, smart homes and buildings, smart agriculture, smart metering, smart supply chain and logistics, and more. With well over 100 million devices connected to networks in 100 countries and growing, LoRa devices are creating a Smarter Planet [9].

Essential features of LoRa Technology as stated by Semtech Corporation, the creator of LoRa technology (following is based on [9]):

1) Long-range – connects devices up to 30 miles apart in rural areas and penetrates dense urban or deep indoor environments.

2) Low power – requires minimal energy, with prolonged battery lifetime of up to 10 years, minimizing battery replacement cost.

3) Geolocation – Enables GPS-free tracking applications, offering unique low power benefits untouched by other technologies.

4) Mobile – Maintains communication with devices in motion without strain on power consumption.

LoRa data rate features a raw maximum data rate of 27 kbps. This is achieved with spreading factor 7 and 500kHz channel. 50 kbps is possible when using FSK modulation instead of LoRa. The higher the spreading factor, the slower is the speed. For example, with the maximum spreading factor of 12, the speeds can drop to 0.3kbps. Another thing to consider is time on air. The bigger the payload, the longer is the time on air [10]. More visualized guide is in the following figure.

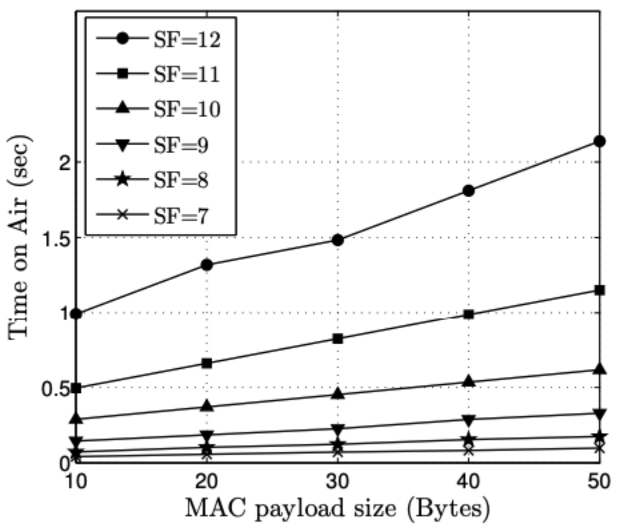

Figure 2. LoRa Time on air with payload sizes [10]

3.2 Data compression

For this project, data compression is crucial as data rates are meagre, and the amount of transferrable data might be quite extensive. When using data compression, we try to save as much bandwidth or shorten the time on air.

Brotli is a generic-purpose lossless compression algorithm that compresses data using a combination of a modern variant of the LZ77 algorithm, Huffman coding and second order context modelling, with a compression ratio comparable to the best currently available general-purpose compression methods. It is similar in speed with deflate but offers more dense compression [11].

Brotli is developed by Google. It came to be after Google employees Jyrki Alakuijala, and Zoltán Szabadka developed it to decrease the size of transmissions of WOFF2 web fonts with the intent to decrease webpage load times. Brotli was a continuation of the development of zopfli, which is a zlib-compatible implementation of the standard gzip and deflate specifications [12].

Brotli achieves 29% greater compression level compared to Deflate's highest level. On the other hand, this comes with higher CPU usage. Same happens on the decompression side [12].

3.3 Data encryption

3.3.1 Symmetric encryption

In symmetric encryption, the same key is used for encryption and decryption. For this to work, both or all parties need to know the key [13]. Nowadays, symmetric encryption is used for bulk encryption and encrypting large chunks of data - for example, databases [14].

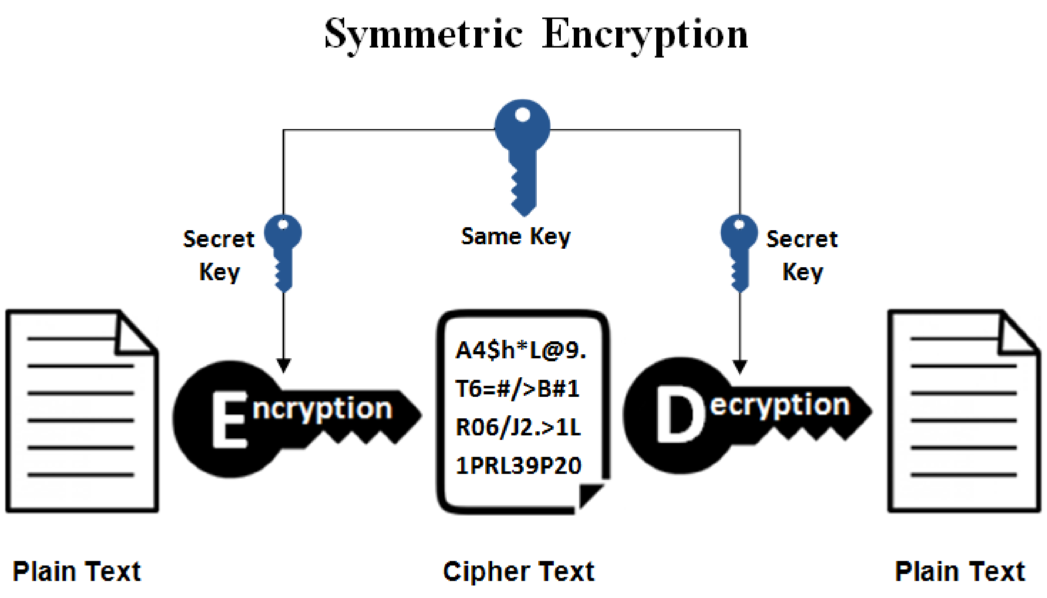

Figure 3. Example of symmetric encryption [15]

Most popular form of symmetric encryption is DES algorithms. It came to be in 1976 in collaboration between IBM and NSA as a federal standard for shared-key encryption. Back then, cryptographers were worried that NSA had implemented weakness in DES algorithms to be able to take advantage of it. Later it was confirmed that this was not the case.

The secretive process that was carried out when choosing DES as a federal standard caused suspicions and concern in the cryptographic communities. This led to choosing

AES as replacement via much improved and entirely public process of proposals and cryptanalysis. This happened in 2001 after a 5-year long contest to replace DES. AES is a version of the Rijndael algorithm designed by Joan Daemen and Vincent Rijmen. AES is also an iterated block cipher, with 10, 12, or 14 rounds for key sizes 128, 192, and 256 bits, respectively.

AES presents high performance symmetric key encryption and decryption. Even after 19 years of observations and research, there are still no substantial attacks against the algorithm that would have been published so far [16].

3.3.2 Asymmetric encryption

In asymmetric encryption, two different keys are used: the sender, who encrypts the data and the recipient who decrypts it. As the name implies, asymmetric encryption is different on each side; the sender and recipient use both different keys. Asymmetric encryption is also known as public-key encryption. It uses public key-private key pairs. Data is encrypted with the public key and decrypted with the private key, or on the contrary, encrypted with the private key and decrypted with public key [17].

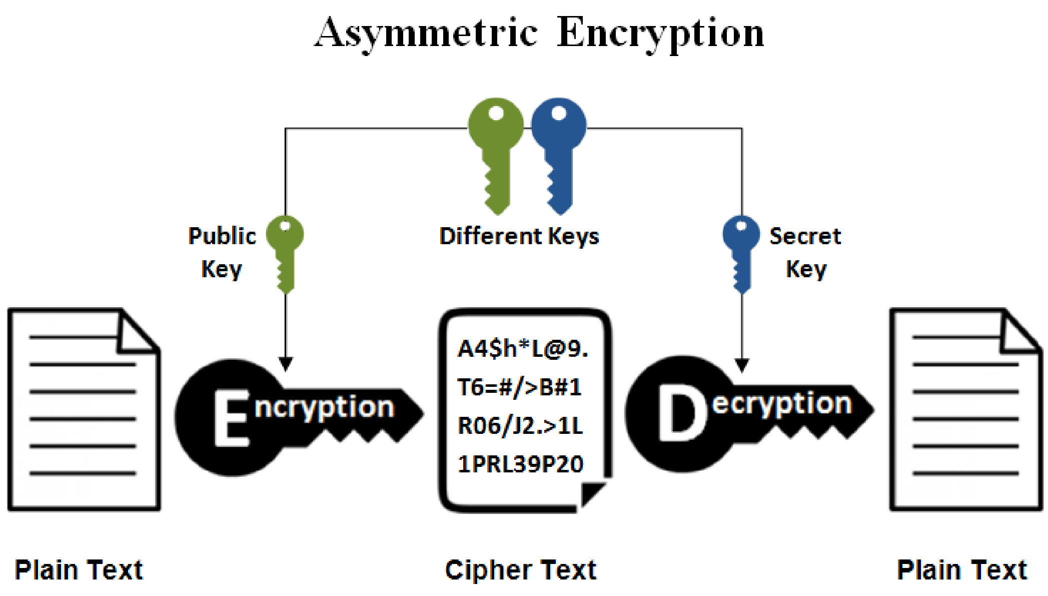

Figure 4. Example of asymmetric encryption [15]

Asymmetric cryptography has two primary use cases: authentication and confidentiality. When using asymmetric cryptography, messages can be signed with a private key, and then anyone with the public key is able to verify that the message was created by someone possessing the corresponding private key. This can be combined with a proof of identity system to know what entity (person or group) actually owns that private key, providing authentication [18].

Asymmetric encryption was proposed first in the paper "New Directions in Cryptography." back in 1977 by Whitfield Diffie and Martin Hellman, researchers at Stanford University. The concept itself had been proposed before by James Ellis several years earlier when he was working for the Government Communications Headquarters, the British intelligence and security organization. The asymmetric algorithm, as described in the Diffie-Hellman paper, implements numbers raised to specific powers to generate decryption keys. Diffie and Hellman had initially collaborated in 1974 to work on solving the problem of key distribution [19].

3.3.3 Comparison

Asymmetric encryption provides higher security but is much slower on the computational side when comparing to symmetric encryption. This is due to the key sizes. For example, maximum bits in AES keys are 256, while RSA supports up to 4096 bits [20].

Furthermore, with asymmetric encryption comes the problem of sharing keys with other counterparties. In casual IP networks, this is not a problem. However, when we have to deal with a non-IP network that has thousands of possible nodes, then sharing everyone's individual keys is a problem. This is even harder in case the key has to be changed during an operation, not before the system is set up in the field.

4 Design of the system

4.1 Network logic

4.1.1 Overview of the network nodes

Planned network will be consisting of two different types of nodes: station node and transmitter node. Station nodes will be doing most of the heavy lifting and transmitter nodes will be just helpers for more manageable tasks that are not meant for syncing data. Majority of the network will consist of station nodes that are intended to be with every bigger unit on the vehicles as devices require power and screens. Transmission nodes are smaller and can act as blue force tracking for smaller units or as sensors on the ground. There are two different types of communication considered. Firstly, heavy communication between station nodes and light communication between the transmitter node and the station node. The following table gives an overview of different symbols that will be used on different figures throughout this thesis.

Table 1. List of symbols that will be used throughout the thesis

Icon	**Explanation**
📶	Station node
📡	Transmitter node
⇄	Heavy communications between station nodes
⟷	Light communication between the transmitter node and the station node

4.1.2 Station node

Station node will be consisting of two Raspberry Pi4's with RAK2245 RPi Hat connected over IP network to additional Raspberry Pi4 running Ubuntu 18.04 that acts as the controller for the entire station node. The following figure shows the possible layout.

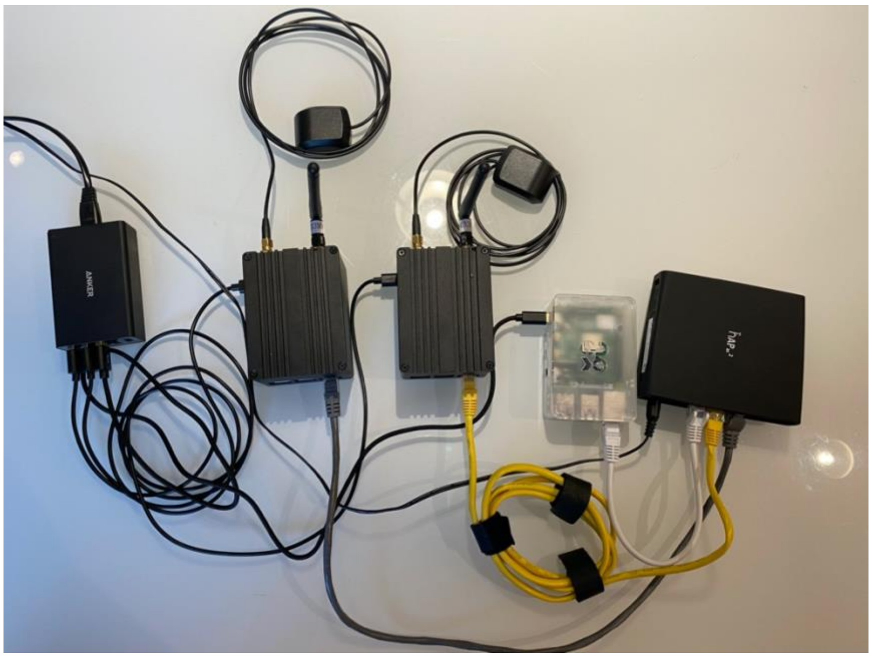

Figure 5. Picture of station node device layout

In the final solution, everything should be better organized and placed in a tight, enclosed space that has power and ethernet connectors embedded. This solution will make setting up and connecting it a lot easier for end-user. Virtual management will be happening over simple browser-based web-app that could.

There will be following services running on the station node:

1) LoRa service on both Raspberry Pi4's with RAK2245. This will transmit received raw data to the controller. Also, it listens for messages coming from the controller. If the controller sends a message to this service, the message will be broadcasted over LoRa.

2) Controller service on the controller node. Controller node receives packets from LoRa services. If both LoRa nodes are listening on the same frequencies, then duplicate data chunks can come in. One instance from the first LoRa service and

second instance from the second LoRa service. Controller node has to be able not to accept duplicates.

3) Management app on the controller node as a web service. This is for end-users to access and manage keys, see network status, debug information and to use chat functionality.

Communication between station nodes will be encrypted using AES256 and pre-shared keys. Managing keys in the local station node would be happening over management web-app that was mentioned before. Communication will be discussed in more details in the following chapters.

4.1.3 Transmitter node

Transmitter nodes will be simple LoRa devices that are preprogrammed and have private key embedded in them. This makes it possible to identify unique transmitter nodes and gives the possibility for end-user to block or ignore some specific devices. The following figure shows the Turtleboard LoRa device by HelTec that was chosen for this project because of simplicity, price and well-documented documentation.

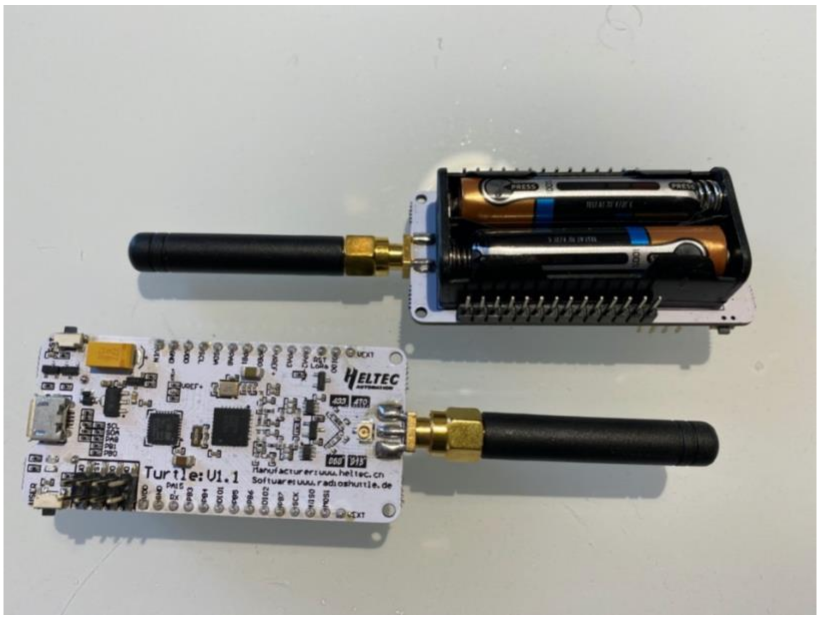

Figure 6. Picture of Turtleboard by HelTec

As this LoRa device can be only either transmitting or listening then, the device will be in a listening mode ready to receive ping messages from station nodes to what it replies with its public key. This public key can be used afterwards by the station node to validate the legitimacy of this device. Previously know the public key can be used to encrypt messages and ask more detailed information about transmitter node - E.g. battery voltage which gives an indication about the battery level. Additionally, self-destruct message can be sent by station node. This makes the transmitter node to delete its private key and renders the device useless if it should fall in the hands of the adversary.

Transmitter node can also be triggered by the pins on the board. For example, there might be a movement sensor that picks up movement by possible adversary. This triggers one pin on the Turtleboard, which in turn will activate transmission mode instead of listening mode after what transmitter node can broadcast that action was triggered, and the person monitoring the station node will get notified. If the location of the transmitter device was known before, then it makes it easy for friendly forces to obtain intel about opposing forces movements or unit can be dispatched to investigate the source of the alert.

Transmitter node can also be connected to a GPS device via serial. This gives the possibility to carry out blue force tracking. That gives an excellent overview of the movement of friendly forces and allows to plan the next stages of the operation better and more accurate as the exact location of the unit is known at all times.

Moreover, any kind of sensor could be developed in the future that could be connected to the transmitter node. This helps friendly forces to have a more thorough overview of what happening and where. Due to this, well-calculated decisions can be made and gives an early warning for possible dangers.

4.2 Communication between nodes

4.2.1 Communication principles

When transmitting data over this low bandwidth non-IP datalinks, there are some conditions that have to be considered at all times. These principles are created by the author taking into account LoRa’s limitations and requirements stated by the Defence League in order for this system to be in accordance with military guidelines.

1) Tininess – Data rates are low, so the amount of transmissions has to be as minimal as possible. This means that every sent message should be compressed as much as possible. (Due to LoRa limitations [10])

2) Uncertainty – As data rates are low, then sending acknowledgement messages takes away even more bandwidth and increases time on air. Mainly all communications should be UDP-style. (Needed because of tininess and silence)

3) Security – Every message has to be encrypted. (Requirement by Defence League)

4) Usability – All broadcast messages should be constructed so that if more than one node receives it, they can get the information as well. Data has to be encrypted with symmetric encryption. Otherwise, only the original recipient will be able to acquire the information. (Taking into account tininess, uncertainty and security)

5) Silence – No broadcasting if not necessary. (Requirement by Defence League)

4.2.2 Message types

Data packets that are sent from one node to another will be considered as messages. Each message that is sent out has to be compressed and encrypted. Messages are divided into different pre-defined message types. This means that one side has to send only data values corresponding to the current message type. This helps to save bandwidth as key values are not transmitted. Different message types are as follows:

1) NOTICE – The aim of this message is to notify others about the existence of this node.

2) PING – Checking if a known node is online if this node has not heard anything from the known node for some amount of time.

3) PONG – Response to PING message by another node.

4) NEW – This node has received a new piece of information over API that has to be broadcasted to other nodes.

5) REQUEST – This node comes to the conclusion that some piece of data is missing and requests it.

6) REPLY – This node got REQUEST message and has the missing piece, so it responds with the requested data.

7) ACTION – This is generated only by transmitter nodes to notify about action at the transmitter node.

8) RATIFY – This is generated only by station nodes in response to transmitter nodes ACTION message.

4.2.3 Synchronization explained

As every station acts as a data consumer and also data producer, then every station is considered equally. The following examples with scenarios will illustrate messaging and data synchronization using the topology in figure 5.

Network has been up for several hours with stations A, B, C, D, E. No station has been broadcasting lately. Timer on station D gets full, and it sends out PING to A, PING to B, PING to C and PING to E. A, B, C and E will respond with PONG. Station A and C now know that station D is up and as they also heard station B PONG message, then they know that station B is up and they do not need to do PING's b themselves. Additionally, station B heard PONG from A and C, so it knows that they are alive.

Sometime later user on station C sends a new chat message. This triggers station C to send out NEW message with chat content. Stations B and D receive it, relay it to chat app and also broadcast the same NEW message. This helps it to reach stations A and E.

After some time, station F comes online. First it sends out NOTICE message. Now station E knows about the existence of station F. Station E sends out PONG message to let station F know that E is in the range. This PONG is again heard also by station D which refreshes its last seen timer on station E.

After a new relation is established and station F does not hear any PONG messages of others, it will send out REQUEST message with a timestamp. In case this message is received, recipients will start sending REPLY messages with all data starting from the newest down to the timestamp that was sent by the requester.

In case the transmitter node is triggered, it sends out ACTION message. If transmitter B sends out ACTION message, it will be heard by stations A, and F. Stations A and F will send back RATIFY message and also NEW message that is converted from ACTION. Transmitter B will receive RATIFY and will not broadcast again for this trigger. However, stations B, D and E will continue relaying the NEW message so that everyone hears it.

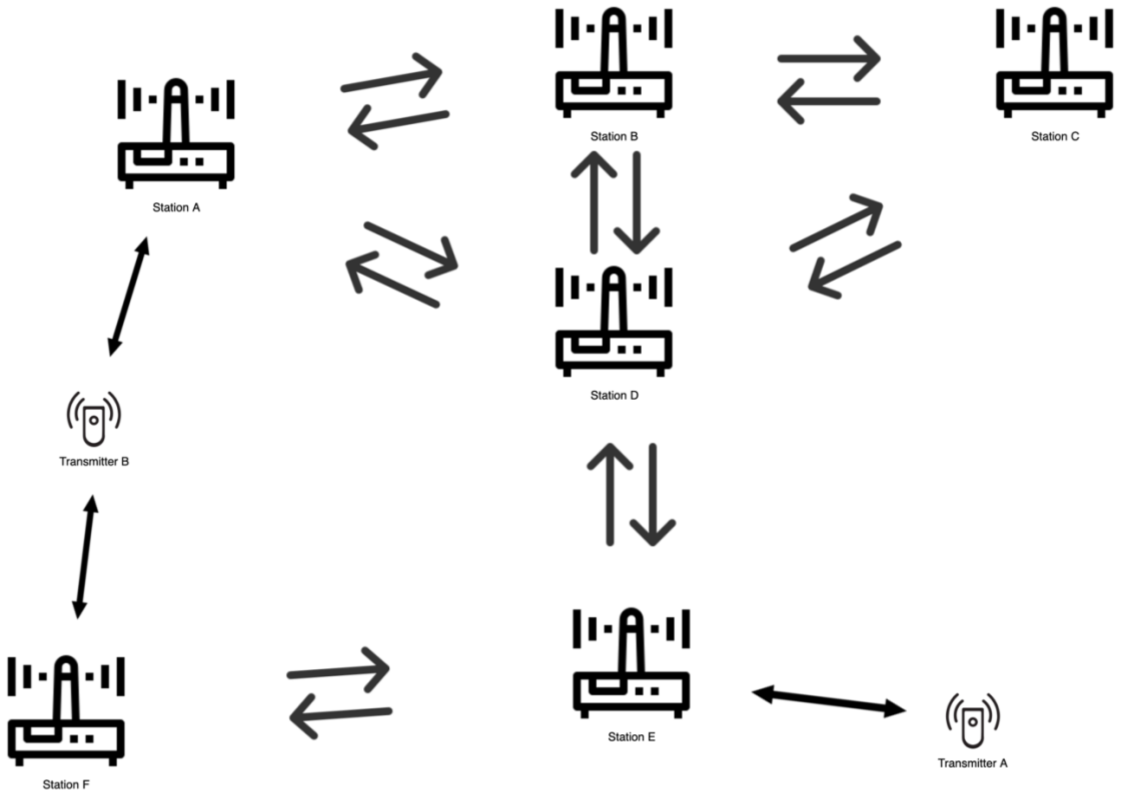

Figure 7. Example network of different nodes

4.3 Service logic

4.3.1 Interfacing

This service is not intended to act as a casual SQL-type database that has its own commands or where different tables with any number of columns could be created. This service cannot read data from user-defined databases or tables. It has its own fixed tables and columns that cannot be changed and to where no additional data should not be inserted by end-user.

There are planned two different methods for inserting and reading the data from data-exchange service. This gives flexibility and allows the service to be used with older software that has their own SQL database and code changes are not possible and also makes it easier to implement this service in new applications or custom-made modules. These options are discussed in more detail in the following chapters.

4.3.2 API

The primary method of interfacing with the data-exchange service is REST API. This allows inserting new data, changing data, reading data, and deleting data. This service allows it to act as a database so that the application does not need to have its own dedicated SQL database.

For every data manipulation type mentioned before will have its own endpoint. Every endpoint can have its own set of parameters. Likewise, every endpoint will be returning its own responses. Parameters used for different endpoints are specified in the following table:

Table 2. List of insert, update, read and delete endpoint parameters

Endpoint	**Parameter key**	**Explanation**
Insert	Key (string)	Will be AES256 key that will be used to encrypt or decrypt data.
Insert	Schema (string)	Which pre-defined schema needs to be used.
Insert	Data (JSON)	Array of JSON objects with data that needs to be inserted.
Update	Key (string)	Will be AES256 key that will be used to encrypt or decrypt data.
Update	UUID (string)	Unique identifier for that specific record
Update	Data (JSON)	JSON object with only the values that are expected to be changed.
Read	Key (string)	Will be AES256 key that will be used to encrypt or decrypt data.
Read	UUID (string)	Unique identifier for that specific record. (optional)

Read	Conditions (JSON)	JSON object of conditions of what data needs to be read from the database. (optional)
Delete	Key (string)	Will be AES256 key that will be used to encrypt or decrypt data.
Delete	UUID (string)	Unique identifier for that specific record.

Values sent back with the response from the server:

Table 3. List of response values from insert, update, read and delete endpoints.

Endpoint	**Key**	**Explanation**
Insert	Success (boolean)	Boolean value if insert operation was successful.
Insert	Error (string)	Error code that is sent only if something went wrong during an insert operation.
Insert	Warning (string)	Warning code that is meant to notify about upcoming deprecations or structure changes.
Insert	IDS (JSON)	JSON array of UUID's that were created.
Update	Success (boolean)	Boolean value if update operation was successful.
Update	Error (string)	Error code that is sent only if something went wrong during the update operation.
Update	Warning (string)	Warning code that is meant to notify about upcoming deprecations or structure changes.
Read	Success (boolean)	Boolean value if read operation was successful.

Read	Error (string)	Error code that is sent only if something went wrong during the read operation.
Read	Warning (string)	Warning code that is meant to notify about upcoming deprecations or structure changes.
Read	Data (JSON)	Returned data.
Delete	Success (boolean)	Boolean value if delete operation was successful.
Delete	Error (string)	Error code that is sent only if something went wrong during a delete operation.
Delete	Warning (string)	Warning code that is meant to notify about upcoming deprecations or structure changes.

There will be additional endpoints that are needed to configure the connection between this service and a custom service. Only one of those is to create and delete new schemas. Schemas are needed so this database service can tell the difference between different entities that external service is interacting with via API. The following table gives an overview of schema creation endpoint parameters:

Table 4. List of parameters for schema creation endpoint

Key	**Explanation**
Name (string)	Name of schema
Schema (JSON)	JSON array of tables, columns that each table has and column datatypes.
Type (LOCAL|PUBLIC)	Defines if this schema needs to be transmitted to other nodes or not.

The following table will give an overview of returning values:

Table 5. List of schema creation response values

Key	**Explanation**
Success (boolean)	Boolean value if schema creation operation was successful.
Error (string)	Error code that is sent only if something went wrong during schema creation operation.
Warning (string)	Warning code that is meant to notify about upcoming deprecations or structure changes.

Schema deletion endpoint takes only in schema name and response is identical to schema creation response. There is no dedicated schema update endpoint. Updating process requires first deleting existing schema and then adding new with the same name.

4.3.3 Interacting with external database

For older or closed source software API instead casual SQL database cannot be used as code cannot be modified or is not reasonable to do so. For this use case, a YAML file can be created that describes database and table schemas, column datatypes, what fields to synchronize and what data to prioritize. Additionally, each YAML file will hold the corresponding key for encryption and decryption, database login credentials, and synchronization time intervals.

With using the time intervals specified in the configuration file, the service will query the database for any changes. If some change is detected, then that data is copied to the service's own database and added to the transmission queue. If transmission with matching data is coming in, then it is added to the external services database that was defined in the configuration file.

This setup is useful, for example, when some service is generating data and needs to transfer it to some central location that is collecting similar data transmissions from several locations and performs some analysis on this data.

5 Designed system analysis

5.1 Speed

In the military environment, the speed of which a piece of information travels from point A to point B is crucial. When having small networks of proposed LoRa based solution, the time of which it takes from the initial sender to reach every other party is expected to be quite modest. Even on bigger networks when there is not much traffic happening, it might take slightly more time to reach others, but it is still considered fast.

On the other hand, when there is a vastly expanded, not bundled network that is under moderate load, then it might take more time. Still, this is an assumption and needs to be confirmed during testing of actual proof-of-concept.

If external service is using configuration files to define synchronization rules to synchronize databases, then additional latency comes from the time interval that is configured for those tables. If the connections time interval is configured in accordance with the data's essence, then speed should not be the issue.

5.2 Reliability

If software will be well-developed and code has gone through extensive testing, then it can be expected that the probability of code side malfunctions will be rather extremely low. As most LoRa devices are expected to work more than ten years in a row even without a battery replacement, then it can be expected that hardware failures are not prone to happen [21]. Additionally, in one station node, there are two LoRa gateway devices which provide needed redundancy in case one device dies due to the software or hardware malfunctions.

When it comes to the transmitter nodes, then they provide no redundancy in case of device or software failure. However, as components in the transmitter device are cheap, assembly is easy, and due to the devices use-case scenarios, they can be considered as throwaway or burner devices. This means that a device breaking or getting lost is not considered as a loss or a risk.

Definitely, additional testing of the devices in this use-case is necessary to be convinced how reliable they are. Additionally, there are a considerable number of vendors that produce LoRa devices and whose products can also be considered as replacements for currently planned devices.

5.3 Scalability

By the theory, this solution can be expected to be scalable and flexible as network sizes can vary, and there are no hard limits. Issues might start arising when we have expanded network with hundreds of station nodes. The speed is in direct correlation with the scalability. The bigger and the more expanded the network, the slower the communications will be.

Another possible issue might arise when several active LoRa networks merge, especially if both networks have had much information exchanged amongst themselves over the last couple of hours. When merging then, both will start broadcasting own data and rebroadcasting received info to other nodes.

On the service's side, scalability should not be the issue as well designed, and optimized code should be able to withstand significant loads. This fact also needs testing when proof-of-concept is ready.

5.4 Security

As long as AES256 algorithm has not been broken by security researchers there seems to be no active threat towards data confidentiality. If one station should fall into enemies' hands, then the protection of the keys should be taken care on the user layer side. Each user should enter keys for themselves, and keys would be encrypted with users own password. This makes capturing of the node useless for the opposing side as contents cannot be decrypted without correct keys.

In the future, additional security functionality could be added to the station nodes by using asymmetric encryption. This would help to implement sending private messages between different nodes that are not neighbours.

5.5 Compatibility

On LoRa connectivity side there should be no problems as LoRa is a standard and no matter the board manufacturer the chip and working logic is still the same. Nevertheless, some vendors may have decided to do some things differently. Those differences might cause some currently unforeseeable problems. After initial proof-of-concept is finished and working with the devices named in this thesis new set of devices should be taken into development and testing process. Additionally, some boards might need different approach or interfacing in order to connect them to LoRa service on Lora Service Raspberry Pi's in station node.

Most anticipated compatibility issues with the main service and external services or databases should be solved with two interfacing options. API can be used for new or custom solutions, as well as for open source solutions. Configuration file-based option works best with third-party services that run on their own dedicated database, or the code cannot be changed in order to use the API.

5.6 Future steps

As stated in several paragraphs before it is necessary to develop actual proof-of-concept using the devices that by the time of writing this thesis were still in transit. Additionally, extensive testing must be carried out. This helps to ensure that devices can withstand the load and usage in this system. Furthermore, testing is vital to root out issues before developing the actual final product.

Summary

The goal of this work was to design proof-of-concept for a system capable of synchronizing message databases over LoRa radio links and to see if this solution could be feasible for future military use.

In the first half of the work, the author explained background information and explored different components that will be playing a crucial role in the planned system. In the second half, the author described and explained the logic of the system and how it will be working. In the last paragraph, the author analyzed if the planned system could accommodate the Estonian Defence League's needs.

Based on the analysis, it was concluded that this system has the potential to succeed, but scalability on a large scale is questionable before actual tests on real devices could be carried out.

In the final analysis, the future action plan was suggested as follows:

1) Developing an actual proof-of-concept using devices that did not reach the author by the time of writing this thesis due to COVID-19.

2) Testing proposed software on proposed devices and measuring and testing its operations.

3) Based on the measurements and testing the final solution should be developed.

References

[1] C. H. Sterling, Military Communications, Santa Barbara, California: ABC-CILIO, Inc, 2008.

[2] “Stalkerzone,” 28 September 2018. [Online]. Available: https://www.stalkerzone.org/russia-will-deploy-electronic-warfare-systems-in-syria-to-counter-high-precision-weapons. [Accessed 17 May 2020].

[3] R. N. McDermott, “Russia's Electronic Warfare Capabilities to 2025,” September 2017. [Online]. Available: https://icds.ee/wp-content/uploads/2018/ICDS_Report_Russias_Electronic_Warfare_to_2025.pdf. [Accessed 5 May 2020].

[4] “JetBrains Rider,” JetBrains s.r.o., 2020. [Online]. Available: https://www.jetbrains.com/rider/. [Accessed 3 May 2020].

[5] “Raspberry Pi,” RASPBERRY PI FOUNDATION, 2020. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/. [Accessed 3 May 2020].

[6] “RAKWireless store,” Shenzhen RAKwireless Technology Co., Ltd., 2020. [Online]. Available: https://store.rakwireless.com/products/rak7244-lpwan-developer-gateway. [Accessed 3 May 2020].

[7] “Heltec Automation,” Heltec Automation, 2020. [Online]. Available: https://heltec.org/project/turtle-board/. [Accessed 3 May 2020].

[8] “Visual Paradigm Online,” Visual Paradigm, 2020. [Online]. Available: https://online.visual-paradigm.com/. [Accessed 6 May 2020].

[9] “What is LoRa?,” Semtech Corporation, 2020. [Online]. Available: https://www.semtech.com/lora/what-is-lora. [Accessed 3 May 2020].

[10] X. V. P. T.-P. B. M. J. M.-S. T. W. Ferran Adelantado, “Understanding the Limists of LoRaWAN,” *IEEE Communications Magazine,* no. January, 2017.

[11] “Google Open Source,” Google LLC, 2020. [Online]. Available: https://opensource.google/projects/brotli. [Accessed 3 May 2020].

[12] “UnusedCSS,” 2020. [Online]. Available: https://unused-css.com/blog/shrinking-your-web-pages-with-brotli/. [Accessed 3 May 2020].

[13] “Cloudflare,” Cloudflare Inc., 2020. [Online]. Available: https://www.cloudflare.com/learning/ssl/what-is-asymmetric-encryption/. [Accessed 3 May 2020].

[14] P. S. &. D. M. Turner, “CRYPTOMAThIC,” Cryptomathic A/S, 18 January 2019. [Online]. Available: https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking. [Accessed 4 May 2020].

[15] “Symmetric vs. Asymmetric Encryption – What are differences?,” SSL2BUY, 2020. [Online]. Available: https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences. [Accessed 17 May 2020].

[16] T. Roeder, “Symmetric-Key Cryptography,” 2010. [Online]. Available: http://www.cs.cornell.edu/courses/cs5430/2010sp/TL03.symmetric.html. [Accessed 4 May 2020].

[17] “Cloudflare,” Cloudflare Inc., [Online]. Available: https://www.cloudflare.com/learning/ssl/what-is-asymmetric-encryption/. [Accessed 4 May 2020].

[18] “Cryptography,” 2020. [Online]. Available: https://cryptography.io/en/latest/hazmat/primitives/asymmetric/. [Accessed 4 May 2020].

[19] M. Rouse, “SearchSecurity,” TechTarget, March 2020. [Online]. Available: https://searchsecurity.techtarget.com/definition/asymmetric-cryptography. [Accessed 4 May 2020].

[20] “Federal Information Processing Standards Publication 197,” 26 November 2001. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf. [Accessed 4 May 2020].

[21] S. Domingo, “How LoRa Can Help You Save The Battery Life of Your IoT Device,” RAKWireless, 1 August 2019. [Online]. Available: https://news.rakwireless.com/how-to-extend-battery-of-your-iot-device/. [Accessed 6 May 2020].