An alternative roaming model in LoRaWAN
Abstract

LoRaWAN is an open networking technology designed for IoT devices that allows wireless data transmission over longer ranges than some other wireless technologies, like Wi-Fi or Bluetooth, for devices that are constrained in terms of size, price, and available power. The current design of roaming among networks in LoRaWAN is heavily inspired by that of mobile networks, as the use of roaming agreements is mandated. Roaming agreements create unnecessary administrative overhead that hinders deployments. A roaming model that is quicker and simpler to deploy could save money for current users, and could even attract new users to the technology. To circumvent the necessity of roaming agreements, a new, scalable and agreement-less roaming model should be proposed. In this thesis project a literature survey is conducted, investigating similar technologies to find hints or inspiration for a new roaming model. It is found that the broker software architecture pattern put in the context of roaming in LoRaWAN suits the requirements quite well, so the new roaming model has been developed based on that. A software simulation has been implemented to gather data regarding the scalability of the model. It has been found that the proposed model is both scalable, and agreement-less.

Keywords: IoT, LPWAN, LoRaWAN, roaming, scalable, agreement-less
6 Results
   6.1 Round one ................................. 20
   6.2 Round Two ................................. 21
   6.3 Round Three ............................... 23

7 Analysis 24

8 Discussion 26

9 Conclusion 27
   9.1 Future work .............................. 27

References 28

A Appendix 1 A
List of Figures

1.1 LoRaWAN Topology ................................................. 2
3.2 Basic Service Sets .................................................. 8
3.3 Distribution System and BSSs ...................................... 9
3.4 Extended Service Set .................................................. 9
3.5 The Tree Structure of DNS ......................................... 11
3.6 Broker Software Architecture Pattern .............................. 11
4.7 Overview of the Proposed Model ................................... 13
4.8 Sequence Diagram of National Roaming ............................ 16
4.9 Sequence Diagram of International Roaming ....................... 17
6.10 Distribution Server CPU used ..................................... 20
6.11 Distribution Server RAM utilization ................................. 21
6.12 Distribution Server CPU used ..................................... 22
6.13 Distribution Server RAM utilization ................................. 22
6.14 Distribution Server CPU used ..................................... 23
6.15 Distribution Server RAM utilization ................................. 24

List of Tables

5.1 Azure VM Details .................................................... 18
6.2 Raw Results of Round One ......................................... 20
6.3 Raw Results of Round Two .......................................... 21
6.4 Raw Results of Round Three ........................................ 23
1 Introduction

This bachelor thesis in computer science was written at Linnaeus University as part of the course 2DV50E, Degree Project at Bachelor level, 15 credits.

1.1 Background

The term Internet of Things (IoT) was first coined by Kevin Ashton in 1999. In an article published in 2009, he clarified what he meant by "Internet of Things" [1]. He explained that computers rely almost entirely on humans for creating the information they process, and that this is limiting because humans are limited in terms of time and accuracy available for creating this information. Therefore, by equipping computers with a capability to gather information and observe the world by themselves, we would “be able to track and count everything, and greatly reduce waste, loss and cost”.

However, the IoT landscape has changed considerably since 1999, and today it is difficult to find a definition that covers everything under the umbrella of IoT. The amount of areas and use-cases covered in IoT have grown immensely, and are still subject to change. The IEEE IoT Technical Community has an ongoing effort with the aim of providing an all-inclusive definition of the IoT domain [2]. Note that they refer to IoT as a domain, and not as a single technology.

A part of this domain is the area of Low-Power Wide-Area Networks (LPWAN) [3]. The goal of LPWAN technologies is to support long range transmissions of small messages sent by simple, usually battery-powered end devices. To enable wide area coverage, most LPWAN technologies operate in the sub-1GHz band, and employ modulation techniques specifically devised with this requirement in mind [4], such as narrowband [5] or spread spectrum [6] modulation. These modulation techniques trade high data throughput for signals that can be decoded potentially tens of kilometers away, that are resilient to jamming or attenuation by obstacles, and that cost minimal power from the devices to produce. As such, LPWAN technologies are suitable for power-constrained devices that communicate using low data-rates, and don’t have stringent latency requirements [4]. There are numerous technologies present in this space. The main subject of this work is LoRaWAN, but SigFox, and NB-IoT are also examples of such technologies.

LoRaWAN is an open networking protocol with oversight from the LoRa Alliance [7]. It is based on the proprietary LoRa modulation technique developed by Semtech. It is defined in three separate specification documents: one deals with the requirements of end devices [8], one deals with the requirements of the backend [9], and another one gives details about parameters that differ based on what region of the world a network is deployed in, such as frequencies used [10].

A LoRaWAN network is deployed in a star-of-stars topology, as described by the specification, and illustrated by Figure 1.1. In the center there are three servers that fulfill three different roles related to the network. The first server is the Network Server (NS) which is responsible for tasks related to messages received and to be sent, such as performing checks, scheduling, and forwarding. The second server is the Join Server (JS) which keeps track of devices activated within a certain network, and the security keys used by them. The third server is the Application Server (AS), which handles the application payloads sent by the devices in the network, generates application payloads that need to be sent to the end devices, and also provides access to the application data to end users. The NS and JS roles are usually fulfilled by the same server. The AS role can be fulfilled together with the other two, but separately as well. Throughout the thesis the combination of NS and JS will be referred to as NS. Between the end devices and the NS are gateways.
(GW). A GW is connected to the NS via a regular IP connection, and to the end devices via LoRa links. The association between end devices and a gateway isn’t as strong as it is in different network technologies, such as when two computers are connected via an UTP cable, or Wi-Fi. A transmission from an end device may be received by multiple GWs, and in this case all of them will forward the message to the NS they are connected to, which is not necessarily the same for all the GWs.

![LoRaWAN Topology](image)

**Figure 1.1: LoRaWAN Topology**

Roaming as a concept has first appeared in telecommunications, when mobile phones started to become widespread, and network operators wanted to extend their services to accommodate them [11]. As defined by the GSM Association, roaming is "the ability of customers to use their mobile phones or other mobile devices outside the geographical coverage area provided by their normal network operator" [12]. Generalizing this definition, we can say that most often, roaming happens when an end device is wirelessly connected to a network, and it leaves this network’s geographical coverage to enter another’s. In this scenario, the original network is called the home network, and the new one is called the visited network. Roaming of devices between service provider networks is bound to roaming agreements in mobile networks [13]. These agreements are used to define the terms of roaming, such as services available to the roaming device, fees, and other implementation details. If a roaming agreement does not exist between the operators of a device’s home network, and the visited network, roaming is not possible for the device.

### 1.2 Related work

Roaming in LoRaWAN is defined in the backend specification [9]. There are two different types of roaming: passive, and handover roaming. During passive roaming all control and application data processing is maintained by the NS of the home network, which the
specification calls the serving NS (sNS). The roaming device uses the GWs of the visited network to send messages to that network’s NS, which the specification calls the forwarding NS (fNS). The fNS simply forwards messages between the sNS and the device. During handover roaming, the home network’s NS is called the home NS (hNS), and the visited network’s NS becomes a serving NS (sNS). Before handover roaming could start, the device needs to complete a so-called rejoin procedure, to allow the sNS to establish some context or session for the particular end device. The sNS still forwards messages to the hNS just like the fNS does in passive roaming, but here the sNS has the added capability to control the radio settings of the roaming devices.

Before any device could roam to any network outside its home, roaming agreements need to be established and configured. According to the specification, a NS should be configured with a roaming agreement for each individual network that could be a target for roaming, allowing or forbidding the different types of roaming individually. Furthermore, a NS should be configured with a roaming agreement that determines which of the devices controlled by it are allowed to roam.

NB-IoT is a LPWAN technology developed by the Third Generation Partnership Program (3GPP), which is also in charge of the standardization of other mobile network technologies, for example 3G and 4G. NB-IoT uses the underlying infrastructure of mobile networks, and licensed bands for all of its communications. It was first specified in Release 13 of the 3GPP in 2016, so it is a rather new technology [14]. Consequently, it is not deployed worldwide yet, and roaming has limited usability. In fact it is not even clear yet what network functions exactly are required to support roaming, although the GSMA is in the process of mapping this with the help of its members [15], [16]. Since NB-IoT is close to 4G in both its origin and its functions, roaming in it is also tied to roaming agreements between network operators, and uses the same interfaces as 4G.

Sigfox is both the name of a LPWAN technology, and the company that oversees it. Sigfox is unique, because it is a single global network deployed only by Sigfox, while being maintained by different operators in each country with presence. This means no matter which country a device connects from, it will connect to the same network. As such, roaming as a concept does not exist in Sigfox. Sigfox is a proprietary technology, and access to the network is sold as a subscription service, therefore only a limited amount of technical details are publicly available [17].

1.3 Problem formulation

It is apparent that the design of LoRaWAN’s roaming model has been heavily inspired by that of mobile networks. However, while roaming agreements are necessary in mobile networks for a multitude of reasons, the same approach creates unnecessary complexity in LoRaWAN networks.

Mobile networks are complex, because they need to be able to keep track of the physical location of the devices to be able to route calls, they need to be robust enough to carry real-time audio without any latency or quality issues to successfully facilitate phone conversations, and they need to meter usage to be able to bill customers precisely.

On the other hand, LoRaWAN networks are simpler, since end devices can move around freely without the need to keep track of their precise location, and the traffic of the networks are small, infrequent messages without stringent latency requirements.

The problem in focus of this work is the problem of the unnecessary overhead of roaming agreements in LoRaWAN. This work contributes to solving this problem by proposing a new model that allows agreement-less, scalable roaming in LoRaWAN.
1.4 Motivation

An improvement of the current approach could be interesting for members of industry, because if roaming would cost less time and effort to set up and maintain, it would indirectly cost less money as well, which is something for-profit organizations always strive for. Another argument for a cheaper process could be that this could make LoRaWAN a more attractive choice over its competitors for customers, which in turn could accelerate the growth of the technology.

1.5 Objectives

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>O1</td>
<td>Investigate technologies that can be used as inspiration for the new model</td>
</tr>
<tr>
<td>O2</td>
<td>Collect the relevant parts of these technologies</td>
</tr>
<tr>
<td>O3</td>
<td>Propose a new model based on the findings of O1 &amp; O2</td>
</tr>
<tr>
<td>O4</td>
<td>Evaluate the proposed model via simulation</td>
</tr>
</tbody>
</table>

The expected result of this research is a new roaming model for LoRaWAN networks. It should allow scalable, agreement-less roaming.

1.6 Scope/Limitation

Limitations of this work include a small number of technologies studied for gathering inspiration, and a lack of diversity in sources of information regarding LoRaWAN. Only a few relevant technologies have been mentioned in this thesis, because it was deemed more preferable to investigate a handful of solid candidates that could potentially serve as a base for the new model than many unrelated technologies. The lack of diversity in information sources regarding LoRaWAN means that real LoRaWAN hardware has been unavailable in the scope of this thesis, so the gathered information regarding LoRaWAN’s operation is based mostly on the specification, and in part on a small number of academic papers.

1.7 Target group

This research may prove relevant to anyone who would want to work with LPWAN technologies in general, or LoRaWAN specifically. Knowledge of basic computer networking concepts is necessary to fully understand this report.

1.8 Outline

In Chapter 2 the method used for discovering a solution to the problem is described. In Chapter 3 the findings of the literature review are presented, and a conclusion is drawn mentioning the best candidate to serve as basis of the new model. In Chapter 4 a new roaming model is proposed, and described in detail. A theoretical use case is also presented. In Chapter 5 the software implementation of the new model, and the environment of the simulations are described. In Chapter 6 the results of this thesis project are shown. In Chapter 7 the results are analyzed and given meaning. In Chapter 8 the results are discussed in terms of the defined problem and objectives. In Chapter 9 the report is concluded, and suggestions for future work are given.
2 Method

To complete objectives O1 and O2, a literature study has been carried out covering technologies that incorporate some sort of roaming or handover procedure, and other technologies used in communication networks that do not define those, but could serve as inspiration nonetheless.

In order to complete O3, the information gathered while accomplishing O1 and O2 has been used to proposed a new model.

In order to prove this model has the desired characteristics, simulations have been conducted where performance metrics have been measured. This covers O4.

2.1 Reliability and Validity

Since at the time of writing this paper there has been no publicly available research on roaming in LPWAN technologies, such works could not be used as references. To overcome this obstacle, the new roaming model has been proposed based on other relevant, widely implemented technologies, and validated via simulation. The supervisors of this thesis have helped in selecting the list of technologies to be researched.
3 Relevant Technologies

In this chapter the results of the literature survey are presented. While each of the mentioned technologies could be studied in a thesis of its own, the focus here has been kept on the relevant parts, be that a roaming/handover procedure, or the overall infrastructure. In the last part of this chapter a conclusion is reached regarding the best candidate to base a new roaming model on.

3.1 4G cellular networks

4G Long Term Evolution (LTE) is the fourth generation cellular network technology that succeeded 3G. Both 3G and 4G use IP as their core, which has allowed video conferencing, HDTV, and IP telephony, while still being mobile.

The requirements for 4G were defined by the International Telecommunication Union (ITU) in 2008. One of these requirements was global roaming capability with simple and efficient handovers [18].

There are two common roaming scenarios in 4G LTE - Home-based routing (S8HR) and Local Breakout. Each scenario has its advantages and disadvantages.

Home-based routing, or S8HR, uses the visited networks’ evolved packet core (EPC) network, while the home network contains an IP multimedia subsystem (IMS). The connection for the roaming device is tunneled back to the home network from the visited network through an IP exchange (IPX) connection. S8HR roaming uses only specific access point names and only within a specific quality-of-service class. The advantage of S8HR is that it is technically easier to deploy compared to the other method and since calls mostly go to the home country, the device can use the same channels as in the home network [19].

Local breakout (LBO) is the other roaming scenario used in 4G LTE, where both the visited network and the home network have an IMS. While using this method call control is served by the visited network and voice control can be served either by home network or the visited network. Both networks establish an interface designation known as network-to-network interface. Not having the data routed to home networks gives advantages such as seamless handover from 4G to 3G and proper handling of emergency calls [18], [19].

Diameter Base Protocol is an authentication, authorization and accounting (AAA) protocol that has evolved from RADIUS, another widely used AAA protocol [20]. This protocol is used in 4G to ensure network access and IP mobility of devices in both local and roaming scenarios as well gathering and forwarding billing data.

Roaming agreements define the conditions and parameters such as authentication, authorization, accounting and billing. Diameter uses those parameters to allow roaming for the network subscribers.

Diameter has four agents that are used to provide scalability, resilience and maintainability [18]. In 4G roaming, a specific type of agent is defined - Diameter Edge Agent (DEA) - which is responsible for IP exchanges (IPX), as well to hide addresses from being exposed to other networks [21]. DEA is considered to be the only point of access into and out of the operator’s network.

On the transport layer, TCP is not used, since it is not able to provide the reliability required for mobile networks. This resulted in the creation of a new protocol called Stream Control Transmission protocol (SCTP) [22].

3G and 4G use SCTP, which was designed for Public Switched Telephone Network (PSTN) to carry data over IP networks, this research does not go into detail about the pro-
tocol, but it is worthy to mention that SCTP has been found to have broader applications [23].

While 4G has a complex and difficult architecture, due to different technologies being used to support all the needs of the end users, the application of underlying protocols or their adaptations in LoRaWAN could allow for simplified roaming. Diameter base protocol is an efficient way of simplifying and automating roaming with an application layer solution. A similar solution, adapted for LoRaWAN could be applied on top of the transport layer.

### 3.2 Media Independent Services

Media Independent Services is, also known as vertical handover, provides a framework that allows higher layers to communicate with lower layers [24], while also providing seamless handover between heterogeneous technologies. All 802.21 compliant devices have a similar structure for Media Independent Handover Function (MIHF). All communication between layers is done through Media Independent Link Service Access Point (MIS LINK SAP), which serves as a common interface for all standard-compliant technologies. The standard defines three types of communication which are called MIS Services [25].

There are three different types of services: event services, which inform the systems of change in state and link. The second type of services is called command services that allow issuing commands from upper layers to lower layers. These services have the ability to be executed locally or remotely. The last type of services is information services, which is used to acquire information about the surrounding area and connectivity, and to supply this information to lower or higher layers, so that the handover is possible.

The commands can be delivered in asynchronous or synchronous mode. Commands generated in lower layers are delivered asynchronously, where query/response type of information is delivered synchronously [24].

With all the described mechanisms, the protocol acts as an intermediary between heterogeneous networks. Application of these features could be valuable to LoRaWAN roaming in detection of nearby sensor hubs to have the ability to send the data to other endpoints.

### 3.3 802.11 WLAN

IEEE standard describes the specifications for the MAC and physical (PHY) layers of a Wireless Local Area Network (WLAN). As it stands in the document, the purpose of this standard is to "provide wireless connectivity for fixed, portable, and moving stations within a local area" [26]. The standard does not describe one specific way of implementation, and therefore its language might seem vague, but for example Wi-Fi technologies are primarily based on this document.

The basic building block of the architecture is the Basic Service Set (BSS), which provides network coverage in a limited physical area. There are four different kinds of BSSs: infrastructure, independent, mesh, and personal. As the name suggests, an infrastructure BSS has some permanent infrastructure in place that facilitates communication, as opposed to the other three, where the participating stations (STAs) are expected to communicate directly with each other. So far we can’t speak of any mobility, beyond a STA’s mobility inside the BSS, as seen in Figure 3.2.

We can build on top of infrastructure BSSs, and extend network coverage beyond the local area. This is achieved by deploying multiple infrastructure BSSs in different
physical locations, and connecting them with a so-called distribution system (DS). The DS uses the distribution system medium (DSM) for passing along any traffic, which is logically different from the wireless medium (WM) used for communication inside a distinct BSS. In order for a BSS to be able to connect to a DS, it must have an access point (AP) in it, which is a STA that has every feature a regular STA has, and has the distribution system access function (DSAF) on top of that. This means that an AP can partake in communications both inside the BSS it is serving, and in the DS with other APs and management entities. The way a DS connects BSSs is illustrated in Figure 3.3.

So far we have multiple distinct BSSs interconnected, but it is possible to take this one step further, and construct what is called an Extended Service Set (ESS). The only difference between and ESS, and multiple BSSs connected by a DS is that in the ESS, every participating BSS has the same Service Set Identifier (SSID). As such, the different physical locations serviced locally by different BSSs appear as one local network, no matter where a STA is inside it. An important concept to note here is that the DS is not part of the ESS, which means that STAs perceive being part of an ESS the same way as they would do in a BSS. The ESS infrastructure is illustrated in Figure 3.4.

The handover procedure that is interesting to us is present in an ESS. We can see that within an ESS there are multiple BSSs, and each BSS has a number of STAs in it. For a STA to be able to communicate in the ESS, it has to go through an association process, which is a service provided by the DS, and it lets the station management entity (SME) know which STA is served by which AP. When a STA wants to change the BSS it is associated with, it can do so using the reassociation process, which is also a service of the DS. Reassociation starts with a STA sending a reassociate request to a SME with some details about itself and the target AP. The SME then evaluates this request, and issues a reassociate response message that is delivered to the requesting STA, with the results of the process. If the results are positive, the STA is now associated with its target AP, if
Figure 3.3: Distribution System and BSSs

Figure 3.4: Extended Service Set
negative, the STA’s association remains unchanged.

We can see that in an 802.11 WLAN a minimum level of infrastructure needs to be present before an endpoint has the chance to move around, but once this minimum is reached the network can be extended to an arbitrary size. Once the minimum is reached, any endpoint can request the management entity for reassociation, and this is only done upon an endpoint’s request.

3.4 Domain Name System

The Domain Name System (DNS) is what allows the World Wide Web, and many other services using the Internet to exist in the form they do today. Computers identify other computers on the Internet using IP addresses, however these are difficult and impractical to memorize for humans. It is much easier for them to memorize names, such as www.wikipedia.org for example. DNS ties these addresses and names together in a globally distributed system.

The exact details of the system were specified in RFC 1034 [27] & RFC 1035 [28] in 1987. Since it is an extensible system, its core has not changed since then, despite the numerous advancements in technology.

DNS is a hierarchical, distributed database. This database contains different types of entries, such as domain names and host names. These names form a tree structure, the domain name space, illustrated in Figure 3.5. Domain names comprise individual labels separated by dots, for example: example.wikipedia.org, where the levels of the hierarchy are represented in a descending order from right to left. We can distinguish five different types of names based on the level they are in the tree. In the descending order of the hierarchy, these are the root that contains all top level domains (TLD), TLDs that collect second level domains based on their common characteristics such as geographical area or purpose, second level domains that are registered to individuals and organizations to use on the Internet, subdomains that are added under a second level domain by its registrant and may serve arbitrary purposes, and individual resource names that identify specific resources [29].

An inspiring concept we can take away from DNS is the hierarchical distribution of data, and responsibilities. The system functions adequately as a whole, but the organizations or entities using it are responsible for the maintenance of accurate records, and the upkeep of some of the infrastructure.

3.5 Broker software architecture pattern

Software architecture is the field of designing software systems to meet certain specifications. Practitioners of this field commonly use patterns. Patterns offer the distilled experiences of skilled architects as reusable solutions to common, recurring problems in software design and implementation [30].

The broker pattern can be used "to structure distributed systems with decoupled components that interact by remote service invocations" [30]. The pattern defines six types of components: clients, servers, brokers, bridges, client-side proxies, and server-side proxies. The aim is to connect clients to services running on servers in a way that does not require the client to know much about the server. This is achieved by having the servers register their services with a broker, and having the clients send their requests to the broker. In a "direct" broker architecture, the only task of the broker is to correctly facilitate the connection of clients and servers, from which point any exchange happens directly between those two entities. In an "indirect" broker architecture, every message travels
Figure 3.5: The Tree Structure of DNS

through the broker. Clients and servers may require proxies on their respective sides, depend-
-ing on the exact implementation scenario. These proxies usually provide a layer of
abstraction. Brokers may be connected via bridges to enable resource sharing between
them. An illustration of this pattern is shown in Figure 3.6.

The idea of an intermediary between the endpoints of the communication could be
interesting, because this introduces a layer of abstraction, and could enable an "opt-in"
-system for taking part in exchanges. This could be useful in removing the complexities
of roaming in LoRaWAN.

3.6 Result

Among the technologies we have studied, only a few implement any kind of handover
procedure. All of these handle handovers in the MAC layer, which requires overhead
in network architecture and signalling capabilities. These are things LoRaWAN cannot
afford to have due to the restraints of the LPWAN operation requirements. Therefore
we have opted to handle roaming at the NS. As the NS already implements the TCP/IP
stack, we can rely on it, and design a model that communicates using a higher layer of the
protocol stack.

Our main goal has been to design a scalable and agreement-less roaming model, there-
fore we have been looking for similar concepts or technologies that could be used as the
basis of our model. The model should also be easily deployable by network operators,
and it should align with the open source mentality of LoRaWAN. We can conclude that not every inspected technology accommodates for these needs, even if at first they seemed like they could be inspiring for us. The two technologies that are the closest to achieving the goals we set for our model are DNS and the broker software architecture pattern. DNS would be a good choice because it defines a certain hierarchy which clearly delimits responsibilities and functions, however it was designed for a specific purpose, and most of the other parts of it would not be easily applied in LoRaWAN while keeping our goals in mind. On the other hand, the broker pattern describes a generic concept that can be customized to fit many scenarios. All these reasons have lead us to adopting a version of the indirect broker software architecture pattern as the base of our roaming model.

By introducing broker entities among potentially hundreds or thousands of networks, the complexity of roaming is significantly reduced. The operator of a LoRaWAN network can choose to opt in to roaming by registering their NS with a broker. The newly registered NS would simply pass on any messages not addressed to itself to the broker, and receive all messages addressed to it but received by different networks from the broker. It is also possible for a registered NS to schedule downlink messages via a broker. Brokers can be deployed by any individual or organization, and access to these brokers can be public, or restricted. Brokers could optionally be configured to share data with one or more other specified brokers to further extend roaming coverage. This results in a robust, flexible, but easy-to-use overall architecture.

In this model, there is only one roaming scenario, which is similar to passive roaming in the current LoRaWAN specification. In this scenario roaming simply means that an end device is already associated with a NS, but a message it sends is picked up by GWs connected to a different NS.
4 Model

Figure 4.7: Overview of the Proposed Model

In the proposed model the existing members of the overall LoRaWAN architecture mostly remain unchanged, however a new entity, a broker, is introduced. In this context the broker entity has been dubbed the Distribution Server. A potential logical topology of the new model is shown in Figure 4.7.

4.1 Distribution Server

As described before, the Distribution Server (DS) provides services that facilitate the roaming among networks. These services should be available via the application layer of the TCP/IP stack.

4.1.1 Registration Service

The registration service handles the association procedure between the current DS, and both NSs, and other DSs. The procedures in these two cases overlap, but have their own specifics as well. The first overlap is that in both cases the only requirement from the clients to be able to initiate association, is to know the IP address of the server. The IP address of the DS should be acquired by the administrator of the client in an out-of-band manner, such as a public website in case of a public DS, or any secure channel in case of a private one. The second overlap is that the clients of the service can achieve two outcomes, either start or stop being associated with the server. Upon receiving a start request, the DS should evaluate it either automatically, or by involving an administrator, and should either grant or deny the request, however a stop request should always be granted. Upon start, all relevant information should be saved by the server, and upon stop this information should be deleted.

Registrations Between DS and NSs

In this scenario the association a NS can request to start or stop refers to participating in roaming via the contacted DS. After the DS grants the start request, the client NS can start to forward all LoRaWAN messages to the DS that are not addressed to itself.
Registration Between DS and Other DSs
In this scenario the association a DS can request to start or stop refers to collaboration between DSs. A DS can both request, and be requested for collaboration. A DS can collaborate with any number of other DSs.

4.1.2 Database Service
The database service is used by the DS running the service to store information related to, and used by its other services. It could be possible to allow trusted parties to query this database, however this could raise some privacy concerns, so in this proposal this service is considered strictly internal.

Data Stored Regarding Roaming NSs
When a NS registers with the DS for roaming, the DS stores information about it. This information must include at least the IP address, and the Network ID (NetID) of the NS. The IP address is necessary because that’s how the DS can reach the NS. The NetID is needed, because a LoRaWAN end device uses this ID to address its NS, so this is how the DS can determine the intended recipient NS of a message.

Data Stored Regarding Collaborating DSs
If an administrator configures the DS to allow collaboration with other DSs, it must store data related to this functionality as well. The minimum amount of data necessary is a list of the collaborating DSs, and which NSs are reachable via which one.

4.1.3 Message Distribution Service
The message distribution service is the one listening for incoming LoRaWAN messages, attempting to find their recipient among the DS’s registered NSs and DSs, and forwarding them to the correct recipient. When the DS processes a message, the first step is always the same: extract the NetID from the message, since this is how LoRaWAN end devices address their NSs. This is the key the DS uses to look for the intended recipient.

Processing Messages Received From Roaming NSs
The second step is to query the database service to see if the NetID matches with that of another NS registered for roaming with the DS. If this is the case, the message distribution service sends the message to the matched NS, and the process terminates. If the first query does not result in success, and the DS does not collaborate with other DSs, the message should be discarded, and the process ends here. If the DS does have collaborating DSs, it should query the database service again to see if any of those serve the intended recipient NS. If yes, the message should be forwarded to this DS, otherwise there are no more options, and the message has to be discarded, and the process ends.

Processing Messages Received From Collaborating DSs
When the sender of an incoming LoRaWAN message is another DS, that means that the remote DS has already determined that the current DS is serving the intended recipient NS of the message. Querying the database service yields the IP address of the correct NS, to which the current distribution service can forward the message.
4.1.4 Information Exchange Service

The information exchange service is only used if the DS has active collaborating DSs. This service is used to send and receive messages between collaborating DSs that contain updates regarding which NSs are reachable via which NS. A DS should send a message to every other DS it is currently collaborating with either periodically, or conditionally. A periodical update would mean similar behaviour to how IP routing protocols operate, where a node sends an update even if nothing has changed in its status, and the update is not a meaningful one. A conditional update mechanism means a DS only updates its collaborating DSs when a change in the NSs it serves occurs. The information contained in these messages should be stored using the database service.

4.1.5 Diagram

An internal block diagram using SysML has been created to illustrate how the described components interact within the DS to make roaming possible. Refer to Appendix A.

4.2 Network Servers

The Network Server's functionality remains largely unchanged. The exception is that they need to implement the client part of the registration, and message distribution services. The only interaction required is by an admin to enter the IP address of a DS, the rest of roaming is handled automatically by the NS.

4.3 Gateways and Devices

The operation of GWs and end devices remain unchanged in our model.

4.4 Use Case Demonstration

To support the proposed model, a use case that demonstrates a possible execution of a real world task is described.

The task the proposed LoRaWAN roaming model is applied to, is international shipment tracking. Specifically the land transport of a package between a warehouse in Stockholm and a warehouse in Oslo is examined.

The scenario is the following. Shipping company A operates in Sweden, and has a country-wide LoRaWAN deployment where each county is covered by a distinct network, and these networks are all connected to the same DS, which thereby provides roaming connectivity to the company’s LoRaWAN end devices wherever they are in the country. They use this LoRaWAN deployment to track valuable shipments by attaching an end device to each parcel, or truck, which sends its location information periodically. Shipping company B operates in Norway, and has an identical LoRaWAN deployment. As these two shipping companies do business with each other often, they decide to link their DSs, so they can track the shipments that run between Sweden and Norway end-to-end.

Before departure, the tracking device is associated with its home NS using one of the options already provided by LoRaWAN. When the tracked shipment departs, the tracking device starts traveling towards the border via Stockholm, Södermanland, Västmanland, Örebro, and Värmland counties, transmitting its location data periodically. Until it leaves Stockholm county, messages are simply travelling within the home network. After that the messages are received by the counties’ NSs, which forward them to the Swedish DS,
which in turn forwards them to the NS of the network covering Stockholm county. This process is illustrated by the sequence diagram in Figure 4.8.

Then it crosses the border and travels through Østfold, Akerhus, and Oslo counties before reaching the final destination. During this time the device transmits its messages, which are received by the Norwegian networks. Here the process is as follows. When a NS receives the message, it passes it on to its DS. The DS doesn’t serve the network the message is addressed to, so it looks at the active collaborating DSs, and finds that the Swedish DS serves the target NS. After this the Norwegian DS sends the message to the Swedish DS, which in turn passes it on to the NS of Stockholm county’s network. This process is illustrated by the sequence diagram in Figure 4.9.
5 Simulation

After having presented the new model in detail, and described its use in a theoretical use case, a more tangible version is also provided in the form of a software simulation. First and foremost, this simulator has been developed with the goal of gathering performance metrics that support claims made regarding the scalability of the model, and only secondarily to provide inspiration for possible future implementations.

5.1 Implementation

The simulator has been implemented in Java, where every element of the new LoRaWAN network model has been implemented as a separate package. The code is available on GitHub [31]. In the code, the real-world operation of a LoRaWAN network and its components has been followed as closely as possible.

5.1.1 Distribution Server

In the Distribution Server code, all the the previously described services of a DS have been implemented, however they are not running as decoupled entities as perhaps suggested before, but as smaller parts of the same program. Since the goal has not been to create a production-ready solution, but to gather relevant data, this has been deemed sufficient.
Beyond the core functionality of the Distribution Server, a small addition has been made which records the amount of processed roaming messages, and prints it to the console once the simulation is over.

5.1.2 Network Server

In the Network Server only a relevant subset of the currently defined NS features has been implemented, which only deals with processing messages. The features necessary for interacting with a DS have been added to this subset. Additionally, a counter has been added to keep track of the amount of incoming messages from the gateways.

5.1.3 Gateways

The functionality of a gateway in this simulator is as defined by the LoRaWAN specification. It merely acts as a forwarder between end devices and a Network Server. The one big difference compared to an actual LoRaWAN gateway is the fact that the connection between end devices and the gateway is not based on LoRa, but on UDP over IP. This is due to the unavailability of LoRaWAN hardware within the scope of this thesis.

5.1.4 End Devices

In order to minimize the resource consumption of the overall simulation, end devices have been implemented as one package that is able to act as an arbitrary amount of LoRaWAN end devices. The ratio of roaming to non-roaming messages produced is configurable. The real world restrictions on duty cycle of an end device have been taken into consideration as well, by implementing a maximum message payload size of 59 bytes, and a static uplink data rate of 31.25 B/s when calculating the delay between transmissions.

5.2 Environment

After implementing the network components, the exact simulation scenario has been decided to include one DS, and two networks served by this DS. This meant seven separate components. As mentioned before, real LoRaWAN equipment has been unavailable, so the simulation has been conducted using virtual machines hosted in the Microsoft Azure cloud, where components have communicated using small UDP messages over the public Internet. In order to prevent wasting resources, the code of the Gateways and Network Servers have been run on the same virtual machine for one network, thereby using only five virtual machines for the seven network components. The specifics of the virtual machines used are presented in table 5.1.

<table>
<thead>
<tr>
<th>Component</th>
<th>Amount</th>
<th>Azure VM size</th>
<th>Virtual CPU cores</th>
<th>Available RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>1</td>
<td>B1ms</td>
<td>1</td>
<td>4 GB</td>
</tr>
<tr>
<td>NS + GW</td>
<td>2</td>
<td>B1s</td>
<td>1</td>
<td>1 GB</td>
</tr>
<tr>
<td>Devices</td>
<td>2</td>
<td>B2s</td>
<td>2</td>
<td>4 GB</td>
</tr>
</tbody>
</table>

Table 5.1: Azure VM Details

These machines have only contained a clean ubuntu 16.04 installation, and the Java 10 SDK. The only exception to this is the Distribution Server where the host-based monitoring tool netdata has also been installed, which has been used to gather usage statistics throughout the simulations.
5.3 Process of simulation

After the environment has been created, the source code has been compiled on the virtual machines. The simulation has been done in three rounds, where in each round ten different scenarios have been simulated for one hour each. In the first scenario there have been one hundred end devices simulated per network, and this number has been increased by one hundred in each sequential scenario, thereby reaching one thousand end devices per network by the tenth scenario. In each of these scenarios, forty percent of the messages generated by the end devices have been roaming messages, and sixty percent have been regular messages. Effectively sixty percent of messages received by Network Server 1 via its Gateway have been addressed to it, and forty percent have been addressed to Network Server 2, thereby making them roaming messages.

5.4 Data gathering

Throughout the simulation several different metrics have been observed. The total amount of incoming messages from a gateway has been recorded by the Network Server implementation, and the total amount of processed roaming messages has been recorded by the Distribution Server implementation. Comparing these numbers provides context regarding the traffic load of the system. Furthermore, the average CPU and RAM utilization has been observed at the Distribution Server. These metrics show how much system resources are being used in the context of the traffic volume.

Starting the simulation and taking periodical measurements has been automated with a shell script. This script made use of the `sar` command to record CPU data, and the `free` command to record RAM data into a text file. The data has been manually compiled into tables after the simulations have finished from the generated text files. To make sure the starting conditions are as close to each other as possible, this script has also scheduled an automatic reboot of the machine after the simulation.
6 Results

The main result of this thesis project is considered to be the new roaming model that has been presented in detail in Chapter 4. The model introduces a new element to the LoRaWAN network model, the Distribution Server, which facilitates the roaming of end devices among networks by utilizing its registration, database, message distribution, and information exchange services.

The secondary results of this thesis project are a small-scale software implementation of the new model, which has been used to conduct experiments by simulating the operation of a LoRaWAN network with different amounts of end devices present, and the data gathered during these simulations. The code can be found on GitHub [31], the simulation data is presented below.

6.1 Round one

<table>
<thead>
<tr>
<th>End devices</th>
<th>Network 1 (#msg received)</th>
<th>Network 2 (#msg received)</th>
<th>Roaming</th>
<th>DS CPU used (%)</th>
<th>DS RAM used (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>3659</td>
<td>3686</td>
<td>4231</td>
<td>2.44</td>
<td>15.40</td>
</tr>
<tr>
<td>200</td>
<td>7289</td>
<td>7407</td>
<td>8581</td>
<td>2.59</td>
<td>15.43</td>
</tr>
<tr>
<td>300</td>
<td>11127</td>
<td>10911</td>
<td>12901</td>
<td>2.58</td>
<td>15.78</td>
</tr>
<tr>
<td>400</td>
<td>14587</td>
<td>14684</td>
<td>16811</td>
<td>2.65</td>
<td>15.54</td>
</tr>
<tr>
<td>500</td>
<td>18307</td>
<td>18140</td>
<td>21249</td>
<td>2.81</td>
<td>15.72</td>
</tr>
<tr>
<td>600</td>
<td>21950</td>
<td>21847</td>
<td>25191</td>
<td>2.92</td>
<td>15.29</td>
</tr>
<tr>
<td>700</td>
<td>25455</td>
<td>25233</td>
<td>29251</td>
<td>3.13</td>
<td>15.62</td>
</tr>
<tr>
<td>800</td>
<td>28998</td>
<td>28861</td>
<td>33503</td>
<td>3.15</td>
<td>15.70</td>
</tr>
<tr>
<td>900</td>
<td>32508</td>
<td>32913</td>
<td>37837</td>
<td>3.25</td>
<td>15.78</td>
</tr>
<tr>
<td>1000</td>
<td>36209</td>
<td>36095</td>
<td>42072</td>
<td>3.4</td>
<td>15.92</td>
</tr>
</tbody>
</table>

Table 6.2: Raw Results of Round One

Figure 6.10: Distribution Server CPU used
6.2 Round Two

<table>
<thead>
<tr>
<th>End devices</th>
<th>Network 1 (#msg received)</th>
<th>Network 2 (#msg received)</th>
<th>Roaming</th>
<th>DS CPU used (%)</th>
<th>DS RAM used (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>3556</td>
<td>3549</td>
<td>4318</td>
<td>2.37</td>
<td>15.50</td>
</tr>
<tr>
<td>200</td>
<td>7133</td>
<td>7111</td>
<td>8572</td>
<td>2.53</td>
<td>15.00</td>
</tr>
<tr>
<td>300</td>
<td>10595</td>
<td>10671</td>
<td>12720</td>
<td>2.62</td>
<td>15.34</td>
</tr>
<tr>
<td>400</td>
<td>14756</td>
<td>14757</td>
<td>16956</td>
<td>2.77</td>
<td>15.13</td>
</tr>
<tr>
<td>500</td>
<td>17979</td>
<td>18214</td>
<td>20923</td>
<td>2.93</td>
<td>16.28</td>
</tr>
<tr>
<td>600</td>
<td>21847</td>
<td>21771</td>
<td>25349</td>
<td>2.95</td>
<td>15.57</td>
</tr>
<tr>
<td>700</td>
<td>25309</td>
<td>25356</td>
<td>29361</td>
<td>3.01</td>
<td>15.27</td>
</tr>
<tr>
<td>800</td>
<td>28856</td>
<td>28965</td>
<td>33482</td>
<td>3.08</td>
<td>15.48</td>
</tr>
<tr>
<td>900</td>
<td>32455</td>
<td>32661</td>
<td>37632</td>
<td>3.17</td>
<td>15.49</td>
</tr>
<tr>
<td>1000</td>
<td>36072</td>
<td>36004</td>
<td>41610</td>
<td>3.37</td>
<td>15.70</td>
</tr>
</tbody>
</table>

Table 6.3: Raw Results of Round Two
Figure 6.12: Distribution Server CPU used

Figure 6.13: Distribution Server RAM utilization
### 6.3 Round Three

The results for Round Three are as follows:

<table>
<thead>
<tr>
<th>End devices (#msg received)</th>
<th>Network 1 (#msg received)</th>
<th>Network 2 (#msg received)</th>
<th>Roaming</th>
<th>DS CPU used (%)</th>
<th>DS RAM used (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>3639</td>
<td>3747</td>
<td>4279</td>
<td>2.48</td>
<td>16.00</td>
</tr>
<tr>
<td>200</td>
<td>7348</td>
<td>7358</td>
<td>8543</td>
<td>2.63</td>
<td>15.85</td>
</tr>
<tr>
<td>300</td>
<td>11030</td>
<td>11088</td>
<td>12789</td>
<td>2.67</td>
<td>15.31</td>
</tr>
<tr>
<td>400</td>
<td>14636</td>
<td>14601</td>
<td>16889</td>
<td>2.79</td>
<td>15.44</td>
</tr>
<tr>
<td>500</td>
<td>18025</td>
<td>18293</td>
<td>20889</td>
<td>3.21</td>
<td>16.34</td>
</tr>
<tr>
<td>600</td>
<td>21744</td>
<td>21685</td>
<td>25196</td>
<td>2.99</td>
<td>15.68</td>
</tr>
<tr>
<td>700</td>
<td>25184</td>
<td>25634</td>
<td>29547</td>
<td>3.02</td>
<td>15.94</td>
</tr>
<tr>
<td>800</td>
<td>28949</td>
<td>28867</td>
<td>33455</td>
<td>3.19</td>
<td>15.80</td>
</tr>
<tr>
<td>900</td>
<td>32495</td>
<td>32519</td>
<td>37516</td>
<td>3.41</td>
<td>15.92</td>
</tr>
<tr>
<td>1000</td>
<td>36168</td>
<td>36215</td>
<td>41951</td>
<td>3.27</td>
<td>15.95</td>
</tr>
</tbody>
</table>

Table 6.4: Raw Results of Round Three

![Distribution Server CPU used](image)

Figure 6.14: Distribution Server CPU used
7 Analysis

In Tables 6.2, 6.3, and 6.4 all of the raw data that has been gathered during the simulations is presented. As it can be seen from the first four columns of each table, the amount of sent and processed messages increases linearly. This is expected, since the amount of devices that are generating them is also increasing linearly throughout the subsequent simulations.

The last two columns contain the data regarding the average CPU load and RAM utilization of the Distribution Server, however these metrics have been broken out into separate graphs to give more attention to them.

In Figures 6.10, 6.12, and 6.14 the average CPU load of the Distribution Server is shown. The CPU utilization shows a linearly increasing trend with some minor deviations in all three rounds, which can also be explained with the linearly increasing nature of the traffic volume in the simulations. It is also apparent that in all the cases the measured average has been in the 2% - 4% interval. This indicates that this application is not particularly CPU-intensive, so even a system with a weaker CPU could act as a Distribution Server with acceptable performance, or a system with an average CPU could act as a Distribution Server for a high number of LoRaWAN networks with good performance.

In Figures 6.11, 6.13, and 6.15 the average RAM utilization of the Distribution Server is shown. As shown in table 5.1, the Distribution Server VM is equipped with 4GB of RAM, so the percentages shown are in relation to that. It is surprising that the data doesn’t show a linear increase as it does in the case of the other metrics, but rather stays within a small interval and shows seemingly random variations inside it. Even though the exact amount of used RAM seems random, the amount is not high. As the data shows, the consumption has remained between 15.00% and 17.00% throughout all the simulations, which translates to 600MB to 680MB of RAM. Multiples of this amount should be readily available in modern computers, especially in servers.

After analyzing the results, it can be concluded that the application’s impact has been
rather low on the system hosting it, and resources left free at the disposal of the Distribution Server have been plentiful. This means that the new LoRaWAN network model presented in this thesis project could scale up to allow roaming among hundreds of networks with server hardware that is not uncommon already today.
8 Discussion

The problem this thesis has set out to solve is the unnecessary overhead of roaming agreements in LoRaWAN. The solution to this problem has been determined to be a new roaming model that is scalable and agreement-less. Initially, the intended steps of defining such a model have been laid out in Chapter 1.5 as a list of objectives. To complete O1 and O2, a literature review has been conducted, and the broker software architecture pattern has been found to be the best candidate for serving as the starting point in developing this new model. O3 has been completed in Chapter 4, where the broker pattern is applied in the relevant context, and the new model is described. Finally, O4 has been completed in Chapters 5, 6, and 7, where the implementation and environment of the simulation have been presented, the results of the simulations have been shown, and the conclusions have been drawn from the gathered data.

As concluded, the new model has been found to fulfill both requirements set for it in the beginning, and as such is a solution to the problem.
9 Conclusion

The current roaming in LoRaWAN is not scalable enough to support the growing IoT field. Considering that no prior research has been done on this topic, the proposed model lays the foundation for future roaming improvements without implementing any changes to the underlying protocols. Additionally, the data gathered during the simulations proves that the model is able to handle a large amount of LoRa devices with a minimal impact to the system.

9.1 Future work

As this work presents a brand new approach to roaming in LoRaWAN, more testing should be done to cover every possible aspect of it. Some of these could include testing for the maximum possible amount of networks and end devices a DS with certain hardware is able to serve, or testing with numerous Distribution Servers collaborating with a certain amount of networks served by them. Once the theory is covered well enough by research and testing, the next step would be to implement the model on real hardware and deploy it in real life to test it in the actual target environments. Some more questions also arise regarding the Distribution Server itself, such as whether it must be a separate entity, or could it be integrated with a Network Server just like the Join Server is currently.

Another idea which might deserve some attention is the Database Service of the Distribution Server. In this thesis it is defined as strictly internal to the Distribution Server, however it could be worth investigating if there is any value in opening it up for querying by trusted parties, and of course the risks that would inevitably accompany this.
References


