SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

TECHNICAL UNIVERSITY OF CRETE

Collection, Decoding, and Recording of data from smart hydrometers for application in a water network

Thesis Defender: Georgios Iniotakis

Committee: Georgios Stavrakakis (Supervisor)

Eftychios Koutroulis

Spiros Papaefthimiou

Chania, July 2024

Abstract

The advent of smart metering technologies has revolutionized water consumption monitoring, enabling more precise and real-time data collection. This thesis explores the implementation of a private LoRaWAN network to manage and monitor sensor data, with a focus on handling an additional encryption layer in data transmission. The primary research question addressed is: How can a secure and efficient private LoRaWAN network be deployed to manage sensor data with additional encryption layers, and how can this data be effectively visualized?

To address this question, a comprehensive methodology was employed, which included setting up a LoRaWAN network server using ChirpStack, deploying sensors with additional encryption, and designing a secure system to handle data decryption and storage. A Flask application was developed to receive and decrypt sensor data, store it in InfluxDB, and visualize it using Grafana. The data decryption was managed by implementing a custom AES decryption algorithm that utilized specific device keys and initialization vectors.

The results demonstrated that the implemented system could successfully decrypt, store, and visualize sensor data in real-time underscoring the potential for advanced metering infrastructure to contribute to more efficient and sustainable water resource management. The Flask application effectively handled incoming data payloads, decrypted the information, and wrote the processed data to InfluxDB. Grafana, configured to read from InfluxDB, provided dynamic and real-time visualizations of the sensor data, allowing for comprehensive monitoring and analysis.

The conclusion of this research indicates that it is feasible to deploy a secure private LoRaWAN network capable of handling additional encryption layers. The integration of ChirpStack, Flask, InfluxDB, and Grafana provides a robust solution for managing and visualizing sensor data. This system can be extended to various applications requiring secure data transmission with LoRaWan technology and real-time monitoring, proving its effectiveness in Internet of Things (IoT) deployments.

Περίληψη

Η έλευση των τεχνολογιών έξυπνης μέτρησης έφερε επανάσταση στην παρακολούθηση της κατανάλωσης νερού, επιτρέποντας τη συλλογή δεδομένων με μεγαλύτερη ακρίβεια και σε πραγματικό χρόνο. Η παρούσα διπλωματική εργασία διερευνά την υλοποίηση ενός ιδιωτικού δικτύου LoRaWAN για τη διαχείριση και παρακολούθηση δεδομένων αισθητήρων, με έμφαση στο χειρισμό ενός πρόσθετου επιπέδου κρυπτογράφησης κατά τη μετάδοση δεδομένων. Το κύριο ερευνητικό ερώτημα που εξετάζεται είναι: Πώς μπορεί να αναπτυχθεί ένα ασφαλές και αποτελεσματικό ιδιωτικό δίκτυο LoRaWAN για τη διαχείριση δεδομένων αισθητήρων με πρόσθετα επίπεδα κρυπτογράφησης και πώς μπορούν να απεικονιστούν αποτελεσματικά τα δεδομένα αυτά;

Για την αντιμετώπιση αυτού του ερωτήματος χρησιμοποιήθηκε μια ολοκληρωμένη μεθοδολογία, η οποία περιελάμβανε τη δημιουργία ενός διακομιστή δικτύου LoRaWAN με χρήση του ChirpStack, την ανάπτυξη αισθητήρων με πρόσθετη κρυπτογράφηση και τον σχεδιασμό ενός ασφαλούς συστήματος για τη διαχείριση της αποκρυπτογράφησης και της αποθήκευσης δεδομένων. Αναπτύχθηκε μια εφαρμογή Flask για τη λήψη και την αποκρυπτογράφηση των δεδομένων των αισθητήρων, την αποθήκευσή τους στην InfluxDB και την οπτικοποίησή τους με τη χρήση του Grafana. Η αποκρυπτογράφηση των δεδομένων διαχειρίστηκε με την εφαρμογή ενός προσαρμοσμένου αλγορίθμου αποκρυπτογράφησης AES που χρησιμοποιούσε συγκεκριμένα κλειδιά συσκευής και διανύσματα αρχικοποίησης.

Τα αποτελέσματα έδειξαν ότι το σύστημα που υλοποιήθηκε μπορούσε να αποκρυπτογραφήσει, να αποθηκεύσει και να οπτικοποιήσει με επιτυχία δεδομένα αισθητήρων σε πραγματικό χρόνο, υπογραμμίζοντας τη δυνατότητα των προηγμένων υποδομών μέτρησης να συμβάλλουν στην αποτελεσματικότερη και βιώσιμη διαχείριση των υδάτινων πόρων.

Acknowledgments

I would like to extend my deepest gratitude to everyone who has supported me throughout the journey of completing this thesis.

First and foremost, I am incredibly grateful to Prof. Papaefthimiou Spiros for giving me the opportunity and the necessary resources to work on this project. His expertise and encouragement have been instrumental in shaping the direction and quality of this research.

I would also like to thank Prof. Koutroulis Eftychios and Prof. Stavrakakis Georgios whose courses, discussions, and study material have profoundly influenced my academic growth and research skills.

I am thankful to Dr. Loukakis Konstantinos, staff of ieesl.tuc.gr for his camaraderie, motivation, and the countless discussions that have enriched my understanding of the subject.

I extend my heartfelt appreciation to my friends and family, whose love and encouragement have been a constant source of strength. To my parents, Petros and Chara, and my friends whose company provided me with the meaning and courage to complete my studies. If they were not that supportive, I would not be here writing these lines. Love you guys.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

Activation by personalization (ABP): Manually provisioning device keys to join a LoRaWAN network. Less flexible, less secure, and less scalable than OTAA, but sometimes useful during demos, to avoid waiting for a downlink window to join a network.

Over-the-Air Activation (OTAA): is the preferred and most secure way to connect a device. Devices perform a join-procedure with the network, during which a dynamic Device Address is assigned and security keys are negotiated with the device.

Adaptive Data Rate (ADR): A mechanism for optimizing data rates, airtime, and energy consumption in the network. Data Rate is a combination of Bandwidth and Spreading Factor which defines how quickly a message is transmitted.

Application Key (AppKey): A device-specific encryption key used during OTAA to derive the AppSKey (in LoRaWAN 1.1x) or both the NwkSKey and AppSKey in LoRaWAN 1.0x. This is usually pre-provisioned by the device manufacturer, but can also be created by the user.

Application Session Key (AppSKey): After activation, this encryption key is used to secure messages that carry a payload.

DevEUI: A 64-bit extended unique identifier for your device. This is programmed by the manufacturer.

Spreading Factor (SF): The transmission speed or Data Rate of a LoRaWAN message, ranging from SF7 (highest Data Rate) to SF12 (lowest Data Rate). Making the spreading factor 1 step lower (from SF10 to SF9) allows you to roughly send the same amount of data used half the time on air. Lowering the spreading factor makes it more difficult for the gateway to receive a transmission, as it will be more sensitive to noise.

Payload: A LoRaWAN packet. Application payloads are decrypted data relevant to the application. Gateway payloads are encrypted and include the application payload plus a frame counter and other LoRaWAN data.

RSSI: The received signal power in milliwatts, measured in dBm. This value is used as a measurement of how well the signal is received. RSSI is used in the ADR feedback loop to determine if a signal is being received with enough power to allow the device to increase its data rate and conserve battery. A high RSSI means the device can reduce transmission power and messages will still be received.

Media Access Control (MAC): LoRaWAN media access control protocols, downloadable from the LoRa Alliance. The Media Access Control version is provided by the manufacturer in a datasheet as the “LoRaWAN version”.

1. INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

The Internet of Things (IoT) has revolutionized the way data is collected and utilized across various industries. IoT devices, equipped with sensors, collect vast amounts of data that are crucial for monitoring, controlling, and decision-making processes. Low-Power Wide Area Networks (LPWANs) such as LoRaWAN have emerged as a popular choice for connecting these devices due to their long-range communication capabilities and low power consumption. However, the security of data transmitted over these networks remains a significant concern. This thesis is motivated by the need to ensure the secure transmission and effective management of sensor data in a private LoRaWAN network, leveraging ChirpStack for network management and additional encryption layers for enhanced security.

In the context of water management, smart water meters have become essential tools for monitoring water usage, detecting leaks, and optimizing resource allocation. The Intelis wSource water meter, a sophisticated IoT-enabled device, offers advanced features for accurate water consumption measurement and data transmission. Integrating this device with the ChirpStack tool, a comprehensive LoRaWAN network management solution, presents an opportunity to explore and enhance the performance of remote water meter monitoring systems.

This study is motivated by the need to address the challenges associated with deploying and managing sensor data securely, especially with additional encryption layers, which pose significant challenges that need to be addressed to ensure data integrity and confidentiality.

1.2 PROBLEM STATEMENT

Despite the advantages offered by smart water meters, the challenge lies in effectively decrypting and interpreting the encrypted payloads transmitted by these devices. The encrypted data, while secure, is not immediately usable for analysis or decision-making. This study addresses the problem of developing a comprehensive method to decrypt these payloads and translate them into meaningful, human-readable data. Without a systematic approach to decode and understand this data, the benefits of smart metering remain underutilized. The primary research problem addressed in this study is how to implement a secure and efficient private LoRaWAN network that can handle sensor data with additional encryption layers and facilitate effective real-time visualization of this data.

1.3 Objectives of the Study

The primary objectives of this study are multifaceted, focusing on the deployment and enhancement of a private LoRaWAN network. Initially, the study aims to deploy a private LoRaWAN network using ChirpStack to efficiently manage sensor data. Building upon this, a robust system will be developed to incorporate additional encryption layers into the data transmission process, ensuring heightened security. Furthermore, a secure decoding mechanism will be designed and implemented to parse decrypted payloads, enabling the extraction of meaningful data and facilitating the storage of sensor data. The study will also focus on real-time visualization of decrypted sensor data by leveraging time-series databases and advanced analytics tools, ensuring that the data is both accessible and actionable. Lastly, the performance of the implemented system will be rigorously evaluated, with efforts made to automate processes where feasible, ensuring a streamlined and efficient operation.

1.4 Research Questions

I. How can a private LoRaWAN network be deployed to reliably manage sensor data with additional encryption layers?

II. What methods can be implemented to securely decrypt and store sensor data?

III. How effective are current tools in visualizing real-time sensor data from a secure LoRaWAN network?

IV. What is the performance of such a system in terms of signal quality?

1.5 Structure of the Thesis

I. **Introduction**: Provides the background, problem statement, objectives, research questions, significance, scope, and structure of the thesis.

II. **Literature Review**: Reviews existing research and technologies related to LoRaWAN.

III. **Methodology**: Describes the methods and tools used for the deployment of the LoRaWAN network, data encryption, decryption, storage, and visualization.

IV. **Results and Discussion**: Presents the findings from the implementation, including performance evaluations.

V. **Conclusion and Future Work**: Summarizes the research findings, discusses the implications, and suggests areas for future research.

2. LITERATURE REVIEW

2.0 INTRODUCTION

This chapter aims to provide a comprehensive review of the literature on IoT technologies, with a particular emphasis on LoRaWAN and its applications in smart water metering. The review begins with an overview of IoT technologies, highlighting the various communication protocols available and their respective advantages and limitations. Following this, an in-depth introduction to LoRaWAN is presented, covering its architecture, benefits, and constraints.

Subsequently, the chapter explores the specific application of LoRaWAN in water meter monitoring, discussing case studies and real-world implementations that demonstrate its effectiveness. Key performance metrics and the impact of environmental factors on these metrics are also examined. The discussion then shifts to the challenges associated with LoRaWAN implementations, such as maintaining signal quality, optimizing Adaptive Data Rate (ADR) settings, and managing network scalability.

Finally, the chapter reviews related work in the field, summarizing key research studies, comparative analyses, and the contributions they have made to advancing our understanding of LoRaWAN-based smart metering systems. This literature review not only contextualizes the current study within the broader field of IoT and smart metering but also identifies gaps and opportunities for further research, laying the groundwork for the subsequent chapters of this thesis.

2.1 OVERVIEW OF IOT TECHNOLOGIES

2.1.1 DEFINITION AND SCOPE OF IOT

The Internet of Things (IoT) refers to a network of physical devices, vehicles, appliances, and other physical objects embedded with sensors, software, and network connectivity, allowing them to collect and share data. IoT devices—also known as "smart objects"—can range from simple "smart home" devices like smart thermostats to wearables like smartwatches and RFID-enabled clothing, to complex industrial machinery and transportation systems. Technologists are even envisioning entire "smart cities" predicated on IoT technologies.

IoT enables these smart devices to communicate with each other and with other internet-enabled devices. Like smartphones and gateways, creating a vast network of interconnected devices that can exchange data and perform various tasks autonomously. This can include:

The potential applications of IoT are vast and varied, and its impact is already being felt across a wide range of industries, including manufacturing, transportation, healthcare, and agriculture. As the number of internet-connected devices continues to grow, IoT is likely to play an increasingly important role in shaping our world. Transforming the way that we live, work, and interact with each other.

In an enterprise context, IoT devices are used to monitor a wide range of parameters such as temperature, humidity, air quality, energy consumption, and machine performance. This data can be analyzed in real time to identify patterns, trends, and anomalies that can help businesses optimize their operations and improve their bottom line.

IoT is important for business for several reasons. Here are the core benefits of IoT:

- Improved efficiency: By using IoT devices to automate and optimize processes, businesses can improve efficiency and productivity. For example, IoT sensors can be used to monitor equipment performance and detect or even resolve potential issues before they cause downtime, reducing maintenance costs and improving uptime.
- Data-driven decision-making: IoT devices generate vast amounts of data that can be used to make better-informed business decisions and new business models. By analyzing this data, businesses can gain insights into customer behavior, market trends, and operational performance, allowing them to make more informed decisions about strategy, product development, and resource allocation.
- Cost-savings: By reducing manual processes and automating repetitive tasks, IoT can help businesses reduce costs and improve profitability. For example, IoT devices can be used to monitor energy usage and optimize consumption, reducing energy costs and improving sustainability.
- Enhanced customer experience: By using IoT technology to gather data about customer behavior, businesses can create more personalized and engaging experiences for their customers. For example, retailers can use IoT sensors to track customer movements in stores and deliver personalized offers based on their behavior.

To make IoT possible, several technologies need to come together. The most important are:

- Sensors and actuators: Sensors are devices that can detect changes in the environment, such as temperature, humidity, light, motion, or pressure. Actuators are devices that can cause physical changes in the environment, such as opening or closing a valve or turning on a motor. These devices are at the heart of IoT, as they allow machines and devices to interact with the physical world. Automation is possible when sensors and actuators work to resolve issues without human intervention.
- Connectivity technologies: To transmit IoT data from sensors and actuators to the cloud, IoT devices need to be connected to the internet. There are several connectivity technologies that are used in IoT, including wifi, Bluetooth, cellular, Zigbee, and LoRaWAN.
- Cloud computing: The cloud is where the vast amounts of data that is generated by IoT devices are stored, processed, and analyzed. Cloud computing platforms provide the infrastructure and tools that are needed to store and analyze this data, as well as to build and deploy IoT applications.
- Big data analytics: To make sense of the vast amounts of data generated by IoT devices, we need to use advanced analytics tools to extract insights and identify patterns.
- Security and privacy technologies: As IoT deployments become more widespread, IoT security and privacy become increasingly important. Technologies such as encryption, access controls, and intrusion detection systems are used to protect IoT devices and the data they generate from cyber threats.

The applications of IoT have spread across various industries offering a plethora of solutions:

1. Healthcare. In the healthcare industry, IoT devices can be used to monitor patients remotely and collect real-time data on their vital signs, such as heart rate, blood pressure, and oxygen saturation. This sensor data can be analyzed to detect patterns and identify potential health issues before they become more serious. IoT devices can also be used to track medical equipment, manage inventory, and monitor medication compliance.
2. Manufacturing. Industrial IoT devices can be used in manufacturing to monitor machine performance, detect equipment failures, and optimize production processes. For example, sensors can be used to monitor the temperature and humidity in a manufacturing facility, ensuring that conditions are optimal for the production of sensitive products. IoT devices can also be used to track inventory, manage supply chains, and monitor the quality of finished products. Industrial IoT is such an expansive new technology space, that it is sometimes referred to by its abbreviation: IIoT (Industrial IoT).
3. Retail. In the retail industry, IoT devices can be used to track customer behavior, monitor inventory levels, and optimize store layouts. For example, sensors can be used to track foot traffic in a store and analyze customer behavior, allowing retailers to optimize product placement and improve the customer experience. IoT devices can also be used to monitor supply chains, track shipments and manage inventory levels.
4. Agriculture. IoT devices can be used in agriculture to monitor soil conditions, weather patterns, and crop growth. For example, sensors can be used to measure the moisture content of soil, ensuring that crops are irrigated at the optimal time. IoT devices can also be used to monitor livestock health, track equipment and manage supply chains. Low-power or solar-powered devices can often be used with minimal oversight in remote locations.
5. Transportation. In the transportation industry, IoT devices can be used to monitor vehicle performance, optimize routes, and track shipments. For example, sensors can be used to monitor the fuel efficiency of connected cars, reducing fuel costs and improving sustainability. IoT devices can also be used to monitor the condition of cargo, ensuring that it arrives at its destination in optimal condition.
6. Smart Homes. Resource optimization can be achieved by installing Devices that monitor and control energy and water consumption in households, leading to a more sustainable and environment-friendly way of living.
7. Environmental Monitoring. IoT-enabled air quality sensors, for instance, continuously measure levels of pollutants and greenhouse gases, offering immediate alerts to both the public and environmental agencies when harmful thresholds are exceeded. Similarly, water quality monitoring systems use a network of sensors to track contaminants, pH levels, and temperature in bodies of water such as rivers, lakes, and reservoirs. These systems can promptly detect pollution events, facilitating rapid responses to prevent environmental damage.

2.1.2 Wireless Communication Technologies

The Internet of Things (IoT) encompasses a vast network of interconnected devices that communicate and exchange data with each other and with centralized servers. The efficiency and reliability of this communication are critical to the successful deployment and operation of IoT systems. Communication protocols play a vital role in enabling these interactions by providing the necessary frameworks and standards for data transmission. These protocols define the rules and conventions for data exchange, ensuring that devices from different manufacturers can interoperate seamlessly.

Selecting the appropriate communication protocol for an IoT application involves considering several factors, including range, data rate, power consumption, and the specific requirements of the use case. The chosen protocol must balance these factors to meet the application's operational needs effectively. A brief overview of the three most known protocols follows.

WIFI

Wi-Fi, or Wireless Fidelity, is a widely used wireless communication protocol based on the IEEE 802.11 standards. It enables devices to connect to a local area network (LAN) and access the internet wirelessly. Wi-Fi operates primarily in the 2.4 GHz and 5 GHz frequency bands, providing high data transfer rates and relatively long-range communication.

Range: Wi-Fi networks typically offer a range of up to 100 meters indoors and 300 meters outdoors. However, the effective range can be significantly reduced by physical obstructions such as walls, floors, and furniture.

Data Rates: Wi-Fi supports high data rates, with the latest standards (such as IEEE 802.11ac and 802.11ax) providing speeds up to several gigabits per second. This makes it suitable for bandwidth-intensive applications such as video streaming, large file transfers, and online gaming.

Power Consumption: Wi-Fi generally consumes more power compared to other IoT communication protocols like Zigbee or LoRaWAN. This higher power consumption can be a limitation for battery-operated IoT devices that require long operational lifetimes.

Zigbee

Zigbee is a specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios. It is based on the IEEE 802.15.4 standard and is designed for low data rate, low power consumption, and secure networking.

Range: Zigbee operates in the industrial, scientific, and medical (ISM) radio bands: 2.4 GHz in most jurisdictions worldwide, though it can also operate in the 868 MHz and 915 MHz bands in Europe and the USA, respectively. The typical communication range of Zigbee is between 10 to 100 meters, depending on the power output and environmental conditions.

Data Rates: Zigbee supports data rates of up to 250 kbps, which is sufficient for many IoT applications that do not require high bandwidth.

Power Consumption: One of the primary advantages of Zigbee is its low power consumption, making it suitable for battery-operated devices that need to operate for long periods without frequent recharging or battery replacement.

5G

5G, the fifth generation of mobile network technology, is designed to provide significantly higher speeds, lower latency, and greater capacity than its predecessors. It is expected to revolutionize various industries by enabling new applications and services that require reliable and high-speed wireless communication. The advent of 5G brings promising advancements for the Internet of Things (IoT), where it can enhance the performance and capabilities of connected devices and systems.

Data rates: 5G supports data rates up to 10 Gbps, enabling real-time data transmission and processing for applications such as video surveillance, augmented reality, and telemedicine, all of which require substantial bandwidth.

Power Consumption: 5G tends to consume way more power than WIFI and Zigbee due to increased data processing and higher frequencies, making it unsuitable for most IoT applications that rely on long-lasting battery devices.

Range: The deployment of 5G infrastructure requires significant investment in new base stations, antennas, and network equipment, which may be cost-prohibitive for some regions.

In conclusion, Zigbee, Wi-Fi, and 5G each have distinct advantages and limitations when used for wireless communication. Zigbee is ideal for low-power, low-data-rate applications in constrained environments, such as smart home automation and industrial control systems, due to its low energy consumption and robust mesh networking capabilities. However, it is less suitable for high-bandwidth applications or scenarios requiring wide coverage. Wi-Fi, with its high data transfer rates and widespread availability, is excellent for applications requiring substantial data exchange, such as video streaming in smart security systems or data-intensive smart appliances. Its high-power consumption and limited range, however, make it unsuitable for battery-operated devices or extensive IoT deployments. 5G, offering ultra-high speeds, low latency, and massive connectivity, is perfect for real-time applications like autonomous vehicles, smart cities, and industrial automation. Nonetheless, the high infrastructure costs and power requirements of 5G make it a poor choice for simple, low-cost, or low-power IoT devices, such as basic environmental sensors or personal fitness trackers.

While each technology's suitability hinges on the specific use case needs and constraints, Low-Power Wide-Area Networks (LPWANs) offer significant advantages over technologies like Zigbee, Wi-Fi, and 5G for many IoT applications, primarily due to their optimized balance between power consumption, range, and cost.

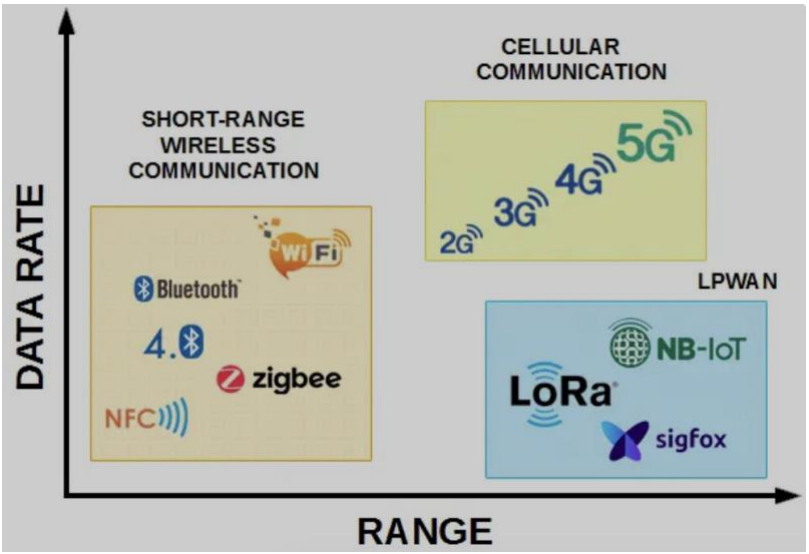


Fig.1 Wireless Communication Technologies

2.1.3 Low Power Wide Area Networks (LPWANs): An Overview

Low Power Wide Area (LPWA) networks represent a novel communication paradigm, which will complement traditional cellular and short-range wireless technologies in addressing diverse requirements of IoT applications. LPWA technologies offer unique sets of features including wide-area connectivity for low power and low data rate devices, not provided by legacy wireless technologies and can make different trade-offs than the traditional technologies prevalent in the IoT landscape such as short-range wireless networks e.g., ZigBee, Bluetooth, Z-Wave, legacy wireless local area networks (WLANs) e.g., Wi-Fi, and cellular networks.

The success of LPWA technologies lies in their ability to offer low-power connectivity to massive number of devices distributed over large geographical areas at an unprecedented low cost. This section describes the techniques LPWA technologies used to achieve these often-conflicting goals. We like to highlight that LPWA technologies share some of the design goals with other wireless technologies. The key objective of LPWA technologies is, however, to achieve a long range with low power consumption and low cost unlike that of the other technologies for which achieving higher data rate, lower latency, and higher reliability may be more important.

A. Long Range

LPWA technologies are designed for a wide area coverage and excellent signal propagation to hard-to-reach indoor places such as basements. Quantitatively, a +20 dB gain over legacy cellular systems is targeted. This allows the end devices to connect to the base stations at a distance ranging from a few to tens of kilometers depending on their deployment environment (rural, urban, etc.). Sub-GHz band and special modulation schemes discussed next, are exploited to achieve this goal.

1) *Use of Sub-1GHz band*: With the exception of a few LPWA technologies (e.g., WEIGHTLESS-W and INGENU), most use the Sub-GHz band, which offers robust and reliable communication at low power budgets. Firstly, compared to the 2.4 GHz band, the lower frequency signals experience less attenuation and multipath fading caused by obstacles and dense surfaces like concrete walls. Secondly, sub-

GHz is less congested than 2.4 GHz, a band used by most popular wireless technologies e.g., Wi-Fi, cordless phones, Bluetooth, ZigBee, and other home appliances. The resulting higher reliability enables long-range and low-power communication. Nevertheless, INGENU's RPMA technology is an exception that still exploits the 2.4 GHz band due to more relaxed spectrum regulations on the radio duty cycle and maximum transmission power in this band across multiple regions.

2) *Modulation Techniques*: LPWA technologies are designed to achieve a link budget of 150 ± 10 dB that enables a range of a few kilometers and tens of kilometers in urban and rural areas respectively. The physical layer compromises on the high data rate and slows down the modulation rate to put more energy in each transmitted bit (or symbol). Due to this reason, the receivers can decode severely attenuated signals correctly. The typical sensitivity of state-of-the-art LPWA receivers reaches as low as -130 dBm. Two classes of modulation techniques namely narrowband and spread spectrum techniques have been adopted by different LPWA technologies.

Narrowband modulation techniques provide a high link budget by encoding the signal in low bandwidth (usually less than 25kHz). By assigning each carrier a very narrow band, these modulation techniques share the overall spectrum very efficiently between multiple links. The noise level experienced inside a single narrowband is also minimal. Therefore, no processing gain through frequency de-spreading is required to decode the signal at the receiver, resulting in a simple and inexpensive transceiver design. NB-IoT and WEIGHTLESS-P are examples of narrowband technologies. A few LPWA technologies squeeze each carrier signal in an ultra-narrow band (UNB) of width as short as 100Hz (e.g., in SIGFOX), further reducing the experienced noise and increasing the number of supported end devices per unit bandwidth. However, the effective data rate for individual end devices decreases as well, thus increasing the amount of time the radio needs to be kept ON. This low data rate in combination with spectrum regulations on sharing underlying bands may limit the maximum size and transmission frequency of data packets, limiting the number of business use cases. SIGFOX, WEIGHTLESS-N, and TELENSA are a few examples of LPWA technologies that use UNB modulation.

spread spectrum techniques spread a narrowband signal over a wider frequency band but with the same power density. The actual transmission is a noise-like signal that is harder to detect by an eavesdropper, more resilient to interference, and robust to jamming attacks. More processing gain is however required on the receiver side to decode the signal that is typically received below the noise floor. Spreading a narrowband signal over a wide band results in less efficient use of the spectrum. However, this problem is typically overcome by the use of multiple orthogonal sequences. As long as multiple end devices use different channels and/or orthogonal sequences, all can be decoded concurrently, resulting in a higher overall network capacity. Different variants of spread spectrum techniques are used by existing standards as discussed in Section III-B and Section III-C. Chirp

Spread Spectrum (CSS) and Direct Sequence Spread Spectrum (DSSS) are used by LoRa and RPMA respectively.

B. Low Power

Low-power operation is a key requirement to tap into the huge business opportunity provided by battery-powered IoT devices. A battery lifetime of 10 years or more with AA or coin cell batteries is desirable to bring the maintenance cost down.

1) *Topology*: While mesh topology has been extensively used to extend the coverage of short-range wireless networks, their high deployment cost is a major disadvantage in connecting a large number of geographically distributed devices. Further, as the traffic is forwarded over multiple hops towards a gateway, some nodes get more congested than others depending on their location or network traffic patterns. Therefore, they deplete their batteries quickly, limiting the overall network lifetime to only a few months to years.

   On the other hand, a very long range of LPWA technologies overcomes these limitations by connecting end devices directly to base stations, obviating the need for the dense and expensive deployments of relays and gateways altogether. The resulting topology is a star that is used extensively in cellular networks and brings huge energy-saving advantages. As opposed to the mesh topology, the devices need not to waste precious energy in busy-listening to other devices that want to relay their traffic through them. An always-on base station provides convenient and quick access when required by the end-devices.

2) *Duty Cycling:* Low power operation is achieved by opportunistically turning off power-hungry components of IoT devices e.g., data transceivers. Radio duty cycling allows LPWA end devices to turn off their transceivers, when not required. Only when the data is to be transmitted or received, the transceiver is turned on.

   LPWA duty cycling mechanisms are adapted based on application, type of power source, and traffic pattern among other factors. If an application needs to transfer the data only over the uplink, the end devices may wake up only when data is ready to be transmitted. In contrast, if downlink transmissions are required as well, the end devices make sure to listen when the base station actually transmits. The end devices achieve this by agreeing on a listening schedule. For example, the end devices may listen for a short duration after their uplink transmissions to receive a reply back. Alternatively, they may wake up at a scheduled time agreed with the base station. For main-powered end devices requiring an ultra-low latency downlink communication, the radio transceiver can stay in an always-on mode. Different LPWA standards such as LORAWAN define multiple classes of the end devices based on their communication needs in uplink or downlink.

In the realm of LPWA technologies, duty cycling the data transceiver is not only a power-saving mechanism but also a legislative requirement. Regional regulations on sharing spectrum may limit the time a single transmitter can occupy to assure its coexistence with other devices sharing the same channel.

Duty cycling can also be extended beyond the transceiver to other hardware components, as explored in the context of many low-power embedded networks. Modular hardware design may provide the ability to choose different operational modes and turn on or off individual hardware components (such as auxiliary components and storage and microcontrollers). By exploiting these power management techniques, LPWA application developers can further reduce power consumption and increase the battery lifetime.

3) *Lightweight Medium Access Control:* Most widely used Medium Access Control (MAC) protocols for cellular networks or short-range wireless networks are too complex for LPWA technologies. For example, cellular networks synchronize the base stations and the user equipment (UE) accurately to benefit from complex MAC schemes that exploit frequency and time diversity. The control overhead of these schemes, while justifiable for powerful cellular UEs, is substantial for the LPWA end devices. Put differently, the control of these MAC protocols may be even more expensive than the short and infrequent machine-type communication of LPWA devices. Further, the very tight synchronization needed by these schemes is difficult to meet by ultra-low-cost end devices having low-quality cheap oscillators. When accessing the spectrum, these devices experience drift in both time and frequency domains, making exclusive access to the shared medium a primary challenge for the competing devices. Due to this reason, simple random-access schemes are more popular for LPWA technologies.

Carrier sense multiple access with collision avoidance (CSMA/CA) is one of the most popular MAC protocols successfully deployed in WLANs and other short-range wireless networks. The number of devices per base station is limited for such networks, keeping the hidden node problem at bay. However, as the number of these devices grows in LPWA networks, carrier sensing becomes less effective and expensive in reliably detecting ongoing transmissions, negatively affecting the network performance. While virtual carrier sensing using the Request to Send/ Clear to Send (RTS/CTS) mechanism is used to overcome this problem, it introduces extra communication overhead over the uplink and the downlink. With massive number of devices, LPWA technologies cannot usually afford this excessive signaling overhead. In addition, link asymmetry, a property of many LPWA technologies today, reduces the practicality of virtual carrier sensing.

Due to these reasons, multiple LPWA technologies such as SIGFOX and LORAWAN resort to the use of ALOHA, a random access MAC protocol in which end devices transmit without doing any carrier sensing. The simplicity of ALOHA is thought to keep the design of the transceiver simple and low-cost. Nevertheless, TDMA-based MAC protocols are also considered by INGENU and NB-IoT to allocate radio

resources more efficiently although at the expense of more complexity and cost for end devices.

4) *Offloading complexity from end devices:* Most technologies simplify the design of end devices by offloading complex tasks to the base stations or to the backend system. The base stations or backend systems have to be more complex to keep the transceiver design for end devices simple. Typically, base stations exploit hardware diversity and are capable of transmitting to and listening from multiple end devices using multiple channels or orthogonal signals simultaneously. This allows end devices to send data using any available channel or orthogonal signal and still reach the base station without the need for expensive signaling to initiate communication. By embedding some intelligence in the backend system, end devices can further benefit from more reliable and energy-efficient last-mile communication.

   A notable example is **LoRaWAN** in which the backend system adapts communication parameters (such as data rate/ modulation parameters) to maintain good uplink and downlink connections. Furthermore, the backend system is also responsible for providing support for end devices to move across multiple base stations and suppress 5 duplicate receptions if any. The choice of keeping complexity at base stations and backend systems, which are fewer in number, enables low-cost and low-power design for many end devices.

C. Low Cost

LPWA technologies adopt several ways to reduce capital expenses and operating expenses for both the end-users and network operators. The low-cost design of end devices is made possible by several techniques some of which are already discussed above. The use of star-type (instead of mesh) connectivity, simple MAC protocols, and techniques to offload complexity from end devices enables manufacturers to design simple and therefore low-cost end devices. Some more techniques, mechanisms, and approaches are the following:

1) *Reduction in hardware complexity*: Compared to cellular and short-range wireless technologies, LPWA transceivers need to process less complex waveforms. It enables them to reduce transceiver footprint, peak data rates, and memory sizes, minimizing the hardware complexity and thus the cost. LPWA chip manufacturers target a large number of connected end devices and can also reduce costs with economies of scale.

2) *Minimum infrastructure:* Traditional wireless and wired technologies suffer from limited range, requiring the dense and therefore expensive deployment of infrastructure (gateways, power lines, relay nodes, etc.). However, a single LPWA base station connects tens of thousands of end devices distributed over several kilometers, significantly reducing the costs for network operators.

3) *Using license-free or owned licensed bands*: The cost to network operators for licensing new spectrum for LPWA technologies conflicts with low-cost deployment, short time-to-market and competitiveness of their subscription offers to customers. Therefore, most LPWA technologies considered deployment in the license-exempt bands including the industrial, scientific, and medical (ISM) band or TV-white spaces. NB-IoT, the LPWA standard from 3GPP, may share the cellular bands already owned by MNOs to avoid additional licensing costs. However, to get better performance, a stand-alone licensed band can be acquired as well, a trend proprietary LPWA technologies may eventually follow to avoid performance degradation due to an increase in the number of connected devices using a shared spectrum.

D. Scalability

The support for massive number of devices sending low traffic volumes is one of the key requirements for LPWA technologies. These technologies should work well with increasing numbers and densities of connected devices. Several techniques are considered to cope with this scalability problem.

1) *Diversity techniques*: To accommodate as many connected devices as possible, efficient exploitation of diversity in channel, time, space, and hardware is vital. Due to the low-power and inexpensive nature of the end devices, much of this is achieved by cooperation from more powerful components in 6 LPWA networks such as base stations and backend systems. LPWA technologies employ multi-channel and multi-antenna communication to parallelize transmissions to and from the connected devices. Further, communication is made resilient to interference by using multiple channels and doing redundant transmissions.

2) *Densification*: To cope with the increased density of the end devices in certain areas, LPWA networks, like traditional cellular networks, will resort to dense deployments of base stations. The problem, however, is to do so without causing too much interference between end devices and densely deployed base stations. Novel densification approaches for LPWA networks need further investigation because existing cellular techniques rely on well-coordinated radio resource management within and between cells, an assumption not true for most LPWA technologies.

3) *Adaptive Channel Selection and Data Rate*: Not only the LPWA systems should scale to the number of connected devices, but individual links should be optimized for reliable and energy-efficient communication. Adapting the modulation schemes, selecting better channels to reach distances reliably, or doing adaptive transmission power control requires efficient monitoring of link qualities and coordination between end devices and the network.

The extent to which adaptive channel selection and modulation is possible depends on the underlying LPWA technology. Different factors such as link asymmetry and maximum allowable radio duty cycle may limit the possibility for very robust adaptive mechanisms. In the cases when the base station is unable to give feedback on the quality of uplink communication and/or inform the end devices to adapt their communication parameters, the end devices resort to a very simplistic mechanism to improve link quality. Such a mechanism includes transmitting the same packet multiple times often on multiple randomly selected channels in the hope that at least one copy reaches the base station successfully. Such mechanisms arguably enhance reliability for this best-effort uplink communication, while keeping the complexity and cost of end devices very low. In the cases when some downlink communication can enable adaptation of uplink parameters, base stations or backend systems can play a vital role in selecting optimal parameters such as channel or optimal data rate to improve reliability and energy efficiency.

Prominent LPWA Technologies

1. SIGFOX

   SIGFOX itself or in partnership with other network operators offers an end-to-end LPWA connectivity solution based on its patented technologies. SIGFOX Network Operators deploy the proprietary base stations equipped with cognitive software-defined radios and connect them to the backend servers using an IP-based network. The end devices connect to these base stations using Binary Phase Shift Keying (BPSK) modulation in an ultra-narrow (100Hz) SUB-GHZ ISM band carrier. By using UNB, SIGFOX utilizes bandwidth efficiently and experiences very low noise levels, resulting in high receiver sensitivity, ultra-low power consumption, and inexpensive antenna design. All these benefits come at the expense of a maximum throughput of only 100 bps. The achieved data rate clearly falls at the lower end of the throughput offered by most other LPWA technologies and thus limits the number of use cases for SIGFOX. Further, SIGFOX initially supported only uplink communication but later evolved into a bidirectional technology, although with a significant link asymmetry. The downlink communication can only precede uplink communication after which the end device should wait to listen for a response from the base station. The number and size of messages over the uplink are limited to 140 12-byte messages per day to conform to the regional regulations on the use of license-free spectrum. Radio access link is asymmetric, allowing transmission of a maximum of only 4 packages of 8-bytes per day over the downlink from the base stations to the end devices. It means that acknowledging every uplink message is not supported.

2. LoRa

   LoRa is a physical layer technology that modulates the signals in the SUB-GHZ ISM band using a proprietary spread spectrum technique developed and commercialized by Semtech Corporation. Bidirectional communication is provided by a special chirp spread spectrum (CSS) technique, which spreads a narrow band input signal over a wider channel

bandwidth. The resulting signal has noise-like properties, making it harder to detect or jam. The processing gain enables resilience to interference and noise.

The transmitter makes the chirp signals vary their frequency over time without changing their phase between adjacent symbols. As long as this frequency change is slow enough so to put higher energy per chirp symbol, distant receivers can decode a severely attenuated signal several dBs below the noise floor. LoRa supports multiple spreading factors (between 7-12) to decide the tradeoff between range and data rate. Higher spreading factors deliver long-range at the expense of lower data rates and vice versa. LoRa also combines Forward Error Correction (FEC) with the spread spectrum technique to further increase the receiver sensitivity. The data rate ranges from 300 bps to 37.5 kbps depending on spreading factor and channel bandwidth. Further, multiple transmissions using different spreading factors can be received simultaneously by a LoRa base station. In essence, multiple spreading factors provide a third degree of diversity after time and frequency.

A special interest group constituted by several commercial and industrial partners dubbed as LoRa Alliance proposed LoRaWAN, an open standard defining architecture, and layers above the LoRa physical layer. We will get into details in the next chapter.

3. NB-IoT

Narrowband Internet of Things (NB-IoT) is an LPWAN technology optimized for low-power, long-range communication, particularly suited for a vast array of IoT devices and services. It operates in licensed frequency bands, ensuring reliable communication with minimal interference. NB-IoT is engineered to provide extensive coverage, penetrating deep into buildings and remote areas where conventional networks may falter. This technology supports a high connection density, enabling the deployment of numerous devices per cell, which is crucial for applications such as smart cities, industrial IoT, and environmental monitoring. The simplicity of device design and the low cost of NB-IoT modules further contribute to its cost-effectiveness, making it a practical choice for widespread IoT adoption.

The low power consumption of NB-IoT devices allows them to operate for several years on a single battery charge, which is ideal for applications requiring long-term deployments without frequent maintenance, such as utility metering and remote sensors. NB-IoT leverages existing cellular infrastructure, offering robust security through established network protocols and ensuring seamless integration within current LTE networks. This facilitates straightforward deployment and management of IoT devices across various sectors, from healthcare and agriculture to smart metering and asset tracking. The technology's ability to deliver reliable and secure communication, coupled with its economic advantages, positions NB-IoT as a pivotal enabler of the rapidly expanding Internet of Things ecosystem.

Comparative Analysis of LoRaWAN with Sigfox and NB-IoT

1. Range and Coverage

LoRaWAN provides an impressive range of up to 15 km in rural areas and 2-5 km in urban environments. Operating in the unlicensed spectrum, it is accessible but may face interference challenges. NB-IoT offers robust coverage through existing cellular infrastructure, suitable for urban settings with a generally lower range than LoRaWAN. Sigfox matches LoRaWAN's range capabilities but can struggle with building penetration and interference.

2. Data Rate and Capacity

LoRaWAN supports variable data rates from 0.3 kbps to 50 kbps, ideal for infrequent, small data transmissions. NB-IoT offers higher data rates up to 250 kbps, accommodating more frequent and larger data packets. Sigfox, limited to very low data rates of up to 100 bps, suffices for simple sensors and status updates.

3. Power Consumption

LoRaWAN is engineered for low power consumption, enabling devices to operate on battery power for several years. NB-IoT, while also optimized for low power usage, typically consumes more power than LoRaWAN due to its cellular nature and higher data rates. Sigfox excels in power efficiency, often surpassing LoRaWAN, making it ideal for long-lasting battery-operated devices.

4. Network Infrastructure and Cost

LoRaWAN necessitates private or public gateways and network servers, with initial setup costs higher due to infrastructure needs, but low operational costs in the unlicensed spectrum. NB-IoT utilizes existing cellular infrastructure, reducing initial setup costs where coverage is available, but with potentially higher operational costs due to carrier fees. Sigfox operates on a low-cost subscription model, suitable for large-scale deployments with minimal infrastructure.

5. Scalability and Interference

LoRaWAN supports millions of devices in a single network, though it can face interference due to the unlicensed spectrum. NB-IoT benefits from the licensed cellular bands' robustness and resistance to interference, ensuring high scalability. Sigfox, while scalable, may encounter interference challenges similar to LoRaWAN.

TECHNICAL SPECIFICATIONS OF VARIOUS LPWA TECHNOLOGIES (European values)

	SIGFOX	LoRaWAN	NB-IoT
Modulation	GFSK - BPSK	DSS with Chirp	QPSK
Band	868 MHz	868 MHz	Licensed LTE bands
Bandwidth	100Hz	125 kHz - 500 kHz	200 kHz
Data rate (uplink)	100 bps	0.3-50 kbps	22 kbps
Range	10-50 km	5-15 km	1-10 km
orthogonal channels	360	8	Single carrier

Link Budget	149-161 dB	153-161 dB	150 dB
messages/day	140	Unlimited	Unlimited
Link symmetry	No	Yes	Yes
Forward error correction	No	Yes	yes
MAC	Unslotted ALOHA	Unslotted ALOHA	LTE-Based
Topology	Star	Star of Stars	Star
ADR	No	Yes	Yes
Payload length	8-12 bytes	Up to 250 bytes	Up to 1600 bytes
Handover	End devices do not join a single base station	End devices do not join a single base station	Supported with seamless handover
Duplex operation	Simplex	Full Duplex	Half duplex
Power Efficiency	Very High	High	Medium
Mobility	Yes	Yes	Limited
Connection Density	Low	Utilized with NB-IoT	1500 devices/km^2
Output Power	14-22 dBm	14-30 dBm	20 dBm
Spectrum Efficiency	Low	High	Medium
Area Traffic Capacity	Low	Depends on gateway	~55k devices per cell
Encryption	AES-128	AES-128	LTE security
Over the air updates	No	Yes	Yes
SLA support	No	No	Yes
Localization	No	Yes	Yes
Interference immunity	Low	Very High	Low

In conclusion, LoRaWAN stands out for its long-range, low power, and adaptive data rate capabilities, making it ideal for specific IoT applications like water metering. Compared to other LPWAN technologies like NB-IoT and Sigfox, LoRaWAN offers a balance of range, power efficiency, and cost-effectiveness. LoRaWAN also stands out for its open standard and flexibility in deployment. Unlike Sigfox, which uses a proprietary protocol, LoRaWAN allows for more customization and is supported by a broad ecosystem of devices and services.

2.2 LoRa and LoRaWAN Explained

As described in the last chapter, Low Power Wide Area Networks (LPWANs) have revolutionized the Internet of Things (IoT) by enabling long-range communication with minimal power consumption, with the prominent LPWAN technologies being LoRa, NB-IoT, and Sigfox, each offering unique features and benefits. This section describes how LoRa works in detail, highlighting the advantages and limitations of LoRaWAN in various IoT applications.

2.2.1 Understanding the LoRa and LoRaWAN differences

LoRa is a communications protocol primarily used for Internet of Things (IoT) devices, able to communicate over ranges of approximately 3 miles in urban areas, and 10 miles in rural areas. Downstream devices communicating over LoRa can operate for years, hibernating while not communicating and pushing a minimal number of bytes upstream at a low power only when necessary.

LoRa uses chirp spread spectrum, using a modulation scheme that encodes data in chirps. The transmitted chirp signals cover a wide frequency range, increasing the signal-to-noise ratio (SNR), the likelihood of the signal being received above the noise floor. This allows for LoRa signals to maintain reliability while being sent at a low power.

The high SNR for low-power transmissions contributes to LoRa's use case for IoT devices needing to operate for long periods without depleting onboard power sources. These low power requirements and high range make it an attractive communication option for IoT deployments in agriculture and heavy industries. Furthermore, these environments are often unfriendly to weak radio signals and lack pre-existing network connectivity over other means.

LoRaWAN is a MAC-layer communication protocol that provides standardized connectivity and security mechanisms for LoRa-based networks. It enables communication between IoT devices and LoRaWAN gateways over distances ranging from a few kilometers in urban areas to over 15 kilometers in rural settings while ensuring interoperability and scalability since it is designed to handle millions of messages from a vast number of devices within a single network. This scalability makes it suitable for large-scale IoT deployments, such as smart cities, industrial monitoring, and agriculture. LoRaWAN incorporates adaptive data rate (ADR) functionality, dynamically optimizing transmission parameters for reliable and efficient communication in varying signal conditions, resulting in very low power consumption, which is crucial for battery life devices.

2.2.2 LoRa Architecture

The LoRa network architecture comprises three key components:

1. End Devices: These are the sensors or actuators that collect and transmit data. They communicate with the network using the LoRa radio interface.
2. Gateways: Gateways receive LoRa signals from end devices and forward them to the network server via standard IP connections. They act as a bridge between the end devices and the network server.
3. Network Server: The network server is responsible for managing the network, including data routing, device authentication, and security.

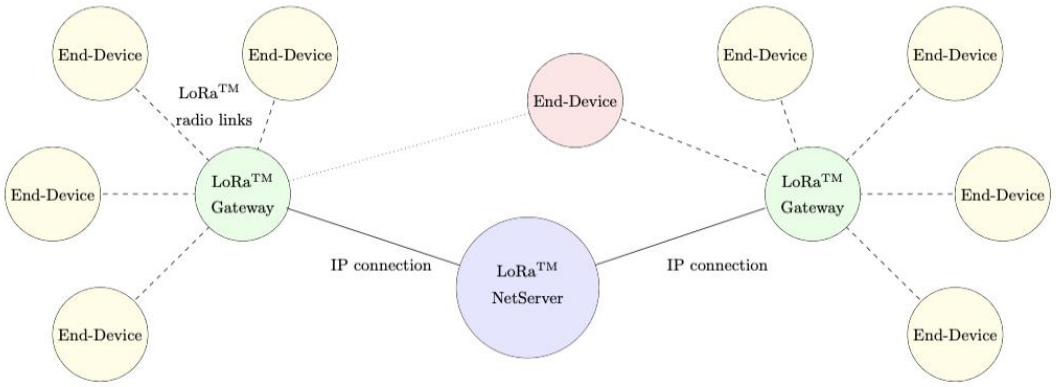


Fig.2 LoRa system architecture

As exemplified in Fig. 1, the network is typically laid out in a star-of-stars topology, where the end devices are connected via single-hop LoRa communication to one or many gateways that, in turn, are connected to a common Network server via standard Internet technologies. The gateways relay messages between end devices and the Network Server, according to the protocol architecture represented in Fig. 2. Interestingly, all the gateways that successfully decode the message sent by an end device will forward the packet to the Network Server by adding some information regarding the quality of the reception. The NetServer will hence reply to the end device by choosing one such gateway, according to some criterion (e.g., best radio connectivity). The gateways are hence totally transparent to the end-devices, which are logically connected directly to the NetServer.

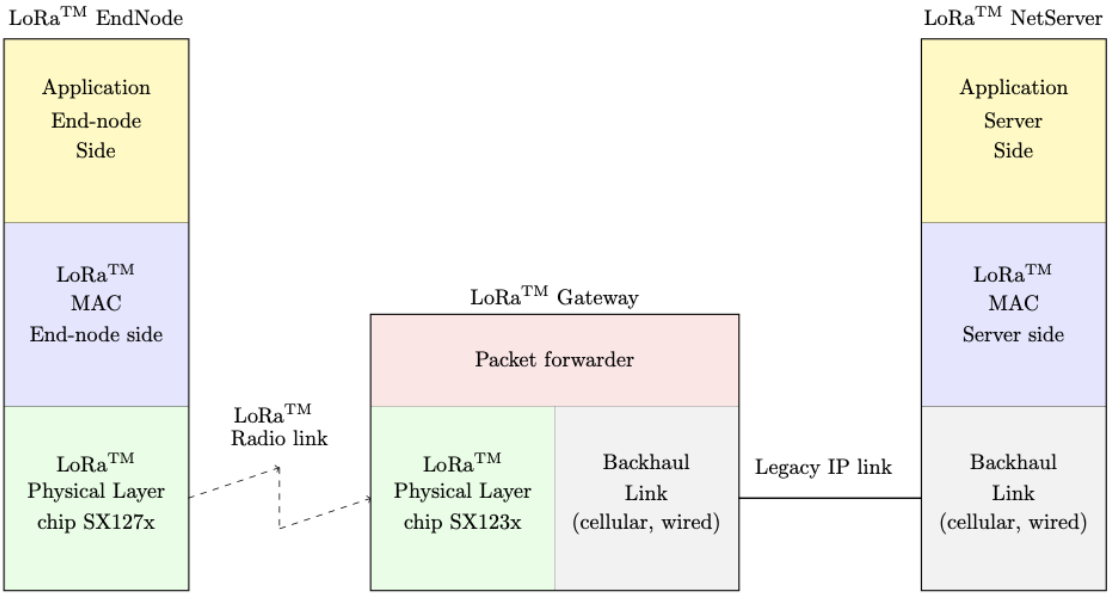


Fig.3 LoRa protocol architecture

2.2.3 LoRa Physical and MAC Layers

The LoRa radio communication is based on a proprietary modulation scheme, which is a derivative of the Chirp Spread Spectrum (CSS) that basically represents symbols as instantaneous changes in the frequency of a chirp. A chirp is defined as a signal whose frequency changes at a fixed rate, which may be constant or exponential. An instantaneous change in the frequency of the chirp, or lack thereof, constitutes a symbol – in essence, symbols are represented as frequency-modulated chirps. The innovation lies in ensuring the phase continuity between different chirp symbols in the preamble part of the physical layer packet, which enables a simpler and more accurate timing and frequency synchronization, without requiring expensive components for generating a stable local clock in the LoRa node. furthermore, the technology supports variable data rates, thus giving the possibility to trade throughput for coverage range, robustness, or energy consumption, while keeping a constant bandwidth.

The LoRa modulation is completely specified by three parameters: the bandwidth BW that, in Europe, usually is 125 kHz or 250 kHz, the spreading factor (the number of encoded bits per symbol) SF ∈ {7, . . . , 12}, which determines the length of the chirp symbol as $T_s = 2^{SF}T_c$, with Tc = 1/BW, and the parameter CR ∈ {0,...,4} which determines the rate of the FEC code as Rate Code = $\frac{4}{4+R}$. A chirp symbol carries SF encoded bits, coming from the interleaver, and every bit carries Rate Code information bits. Therefore, the effective data rate of an infinitely long payload is given by:

$$\text{Data Rate} = \text{SF}\frac{\text{Rate Code}}{\frac{2^{SF}}{BW}} \text{ bit/s}$$

Considering that each packet is prepended with a preamble to allow for frequency and timing synchronization at the receiver, the actual data rate ranges approximately from 0.3 kbps to 11 kbps, with BW = 250 kHz. However, the system capacity is larger because the receiver can detect multiple simultaneous transmissions from different nodes by exploiting the orthogonality of the spreading sequences used by LoRa.

The MAC layer defined by the LoRa Alliance is called LoRaWAN and it is basically an ALOHA protocol controlled primarily by the LoRa Network Server. A distinguishing feature of the LoRa MAC is the adaptive data rate, which allows the Network Server to adapt the transmit rate of an end-device by changing the SF index, to find the best trade-off between energy efficiency and link robustness. Another important feature is the strong security mechanisms that entail a network key and an application key, which are set up through an over-the-air activation procedure, as well as activation by personalization procedure (where the security parameters are set into the device at production time).

2.2.4 Advantages and Limitations of LoRaWAN

Long-Range Communication

LoRaWAN is designed to provide long-range communication capabilities, making it suitable for wide-area networks. It can achieve communication distances of up to 15 kilometers in rural areas and several kilometers in urban environments. This makes LoRaWAN ideal for applications where devices are spread out over large geographical areas, such as agriculture, smart cities, and environmental monitoring.

Low Power Consumption

One of the most significant advantages of LoRaWAN is its low power consumption. Devices using LoRaWAN can operate on battery power for several years without the need for frequent replacements or recharging. This is particularly beneficial for applications like remote sensing and metering, where devices are deployed in hard-to-reach or maintenance-intensive locations.

Robustness and Reliability

LoRaWAN uses a chirp spread spectrum modulation technique, which makes it highly resistant to interference and multipath fading. This enhances the robustness and reliability of the network, ensuring consistent communication even in environments with high levels of interference.

Scalability

LoRaWAN networks are highly scalable, supporting thousands of devices in a single network. This scalability is facilitated by the star-of-stars topology, where multiple end devices communicate with gateways that forward data to a central network server. This architecture allows for efficient network management and easy expansion as the number of connected devices grows.

Cost-Effectiveness

The infrastructure required for LoRaWAN networks is relatively simple and cost-effective compared to other communication technologies. The use of unlicensed frequency bands also eliminates the need for costly spectrum licenses, further reducing deployment and operational costs. This makes LoRaWAN a cost-effective solution for large-scale IoT deployments.

Flexible Data Rates

LoRaWAN supports adaptive data rates (ADR), which dynamically adjust the data rate based on the signal quality and network conditions. This flexibility ensures optimal use of network resources, balancing the trade-off between communication range and data rate to maintain reliable connections.

Limited Data Throughput

While LoRaWAN is excellent for low-power, long-range communication, it is not designed for high data throughput. The data rates supported by LoRaWAN are relatively low, ranging from 0.3 kbps to 50 kbps. This limitation makes it unsuitable for applications that require high bandwidth or real-time data transmission, such as video streaming or high-frequency sensor data.

Duty Cycle Restrictions

To comply with regulatory requirements, LoRaWAN devices operating in unlicensed frequency bands are subject to duty cycle limitations. This means that devices can only transmit for a small fraction of the time (e.g., 1% of the time in Europe). These restrictions can limit the amount of data that can be sent and the frequency of transmissions, which might impact applications requiring frequent updates.

Network Congestion

In dense urban areas or environments with a high concentration of LoRaWAN devices, network congestion can become a significant issue. The limited number of available channels and the potential for overlapping transmissions can lead to increased packet loss and reduced network performance.

Interference and Signal Attenuation

Although LoRaWAN is designed to be robust against interference, it can still be affected by physical obstacles and environmental factors. Buildings, trees, and other structures can attenuate the signal, reducing the effective communication range and reliability. Additionally, operating in unlicensed bands means LoRaWAN devices may experience interference from other wireless technologies.

Security Concerns

While LoRaWAN includes security features such as encryption and authentication, there are still potential vulnerabilities that need to be addressed. The use of shared keys, the potential for physical tampering of devices, and susceptibility to certain types of attacks (e.g., replay attacks) require careful consideration and implementation of additional security measures.

Latency

LoRaWAN is not designed for low-latency communication. The time it takes for a message to travel from the end device to the network server and back can be significant, especially when using confirmed messages or downlink communications. This latency can be a limitation for applications requiring real-time responsiveness.

2.3 APPLICATIONS OF LORAWAN IN WATER METER MONITORING

2.3.1 SMART METERING: AN OVERVIEW

Smart metering refers to the technology of using advanced meters that can measure and record the consumption of utilities, such as electricity, gas, and water, in real-time or near real-time. Unlike traditional meters, which require manual readings, smart meters automatically send usage data to the utility provider, enabling more efficient monitoring, billing, and management. Smart meters are equipped with digital displays and communication capabilities, allowing for two-way communication between the meter and the central system. This communication can be facilitated through various technologies, including cellular networks, RF mesh, power line communication, and more recently, Low Power Wide Area Networks (LPWAN) like LoRaWAN.

The benefits of smart metering systems include Accurate and Timely Data Collection. Smart meters provide precise and real-time data, reducing the errors associated with manual readings and estimated billing. This accuracy helps in ensuring that consumers are billed correctly for their actual usage.

Smart metering also enhances Operational Efficiency. Automated data collection reduces the need for manual meter readings, cutting down on labor costs and minimizing human errors. This efficiency also allows utility companies to allocate resources more effectively.

Another major benefit is the Improved Demand Management. With access to detailed consumption data, utility providers can better understand usage patterns and manage demand. This capability is particularly valuable during peak periods, helping to avoid outages and optimize the distribution of resources.

Furthermore, In the case of water meters, smart metering systems can help detect leaks and unusual consumption patterns early, preventing significant water loss and damage. This proactive approach contributes to better resource conservation.

Lastly, smart meters enable consumers to monitor their usage in real-time, often through online portals or mobile applications. This transparency encourages more responsible usage behavior and helps consumers identify opportunities for energy or water savings. After defining the benefits of smart metering and LoRaWAN technology, let's discuss the role it could have in smart water metering and how we could leverage it to build applications.

2.3.2 LORAWAN IN WATER METERING

The integration of LoRaWAN technology in water metering systems has gathered significant attention due to its potential to revolutionize water management practices. LoRaWAN, with its long-range communication capabilities and low power consumption, is particularly well-suited for the requirements of smart water metering. This section explores the application of LoRaWAN in water metering, highlighting case studies and real-world applications, and discussing the specific advantages it offers.

Several municipalities and utility companies around the world have successfully implemented LoRaWAN-based water metering systems. These case studies demonstrate the practical benefits and challenges of deploying such technologies in diverse environments.

- Veolia, a global leader in optimized resource management, deployed LoRaWAN-enabled water meters in several French cities. The project aimed to enhance water management efficiency and reduce water losses. Thousands of LoRaWAN-compatible water meters were installed across urban and rural areas, connected to a centralized LoRaWAN network. The initiative led to significant improvements in leak detection, meter reading accuracy, and overall water management. Veolia reported a substantial reduction in non-revenue water and improved customer satisfaction.
- San Francisco Public Utilities Commission implemented a LoRaWAN-based water metering system to modernize its infrastructure and improve water conservation efforts. The system included the deployment of advanced metering infrastructure (AMI) with LoRaWAN-enabled meters and gateways across the city. The project achieved enhanced data collection capabilities, allowing for real-time monitoring of water usage. SFPUC observed better demand management and increased operational efficiency.
- The Bangalore Water Supply and Sewerage Board undertook a pilot project to implement smart water meters using LoRaWAN technology to address water scarcity issues. The project involved the installation of LoRaWAN-enabled meters in selected zones, coupled with data analytics tools to monitor usage patterns. The pilot demonstrated improved accuracy in billing, reduced water wastage, and better-informed decision-making for water resource management.

2.3.3 Key Metrics and Performance Indicators

The performance of LoRaWAN-based water metering systems is critical for ensuring reliable data transmission, efficient network operation, and accurate monitoring. Several key metrics and performance indicators are used to evaluate these systems. Understanding and optimizing these metrics can significantly enhance the overall performance and reliability of the network.

I. Received Signal Strength Indicator (RSSI)

RSSI is a measure of the power level that a wireless receiver receives from a transmission. In LoRaWAN networks, RSSI is a crucial metric for evaluating the quality of the received signal. It is measured in decibels milliwatt (dBm) and indicates how well a signal can be detected by the receiver.

Importance: High RSSI values indicate strong signals, which are essential for maintaining a reliable connection between the end device (water meter) and the gateway.

Factors Affecting RSSI: Distance between the transmitter and receiver, physical obstructions (buildings, walls), environmental conditions (weather, terrain), and interference from other electronic devices.

Optimization: Placing gateways in strategic locations to minimize obstructions and interference can help improve RSSI values. Using higher gain antennas can also enhance signal strength.

II. Signal-to-noise ratio (SNR)

SNR is the ratio of the power of the desired signal to the power of background noise. It is measured in decibels (dB) and reflects the quality of the signal received by the gateway.

Importance: A high SNR indicates a clear and strong signal, which reduces the likelihood of data corruption and packet loss. It is essential for reliable data transmission and accurate readings from the water meter.

Factors Affecting SNR: Similar to RSSI, SNR is influenced by distance, obstructions, environmental conditions, and interference. Additionally, noise from other electronic devices and networks can degrade SNR.

Optimization: Reducing interference and noise sources, such as using proper shielding and placing gateways away from noisy environments, can improve SNR. Implementing directional antennas can also focus the signal, enhancing SNR.

III. Packet Delivery Rate (PDR)

PDR is the ratio of successfully received packets to the total number of packets sent by the end device. It is a critical metric for assessing the reliability and efficiency of the LoRaWAN network.

Importance: A high PDR indicates that most of the data packets are successfully transmitted and received, ensuring accurate and complete data collection from the water meters.

Factors Affecting PDR: Signal quality (RSSI and SNR), network congestion, interference, and the physical environment can impact PDR. The configuration of data rates and Adaptive Data Rate (ADR) settings also play a significant role.

Optimization: Ensuring optimal RSSI and SNR, minimizing interference, and fine-tuning ADR settings can improve PDR. Network management strategies, such as load balancing and efficient channel allocation, can also enhance PDR.

IV. Data Rate and Adaptive Data Rate (ADR)

Data rate refers to the speed at which data is transmitted between the end device and the gateway. ADR is a feature in LoRaWAN that automatically adjusts the data rate based on the network conditions and signal quality.

Importance: Optimizing the data rate ensures efficient use of network resources, reduces power consumption of the end devices, and maintains reliable communication.

Factors Affecting Data Rate: Signal strength (RSSI and SNR), distance between the device and gateway, and network congestion influence the data rate. Higher data rates require stronger signals and lower noise levels.

Optimization: Implementing ADR to dynamically adjust data rates based on real-time network conditions can optimize performance. Ensuring a balance between high data rates and reliable communication by considering the environment and network load is crucial.

V. Latency

Latency is the time taken for a packet of data to travel from the end device to the application server and back. It is an essential metric for real-time applications and monitoring systems.

Importance: Low latency is crucial for real-time data monitoring and immediate response actions, such as alerts and automated controls.

Factors Affecting Latency: Network configuration, routing paths, gateway processing times, and server response times impact latency.

Optimization: Optimizing network architecture, reducing the number of hops in data transmission, and ensuring efficient processing at gateways and servers can minimize latency.

VI. Battery Life of End Devices

For battery-operated water meters, the battery life is a critical performance indicator. Efficient use of power ensures long-term operation without frequent battery replacements.

Importance: Prolonged battery life reduces maintenance costs and ensures continuous monitoring.

Factors Affecting Battery Life: Transmission frequency, data rate, power settings, and environmental conditions.

Optimization: Implementing low-power transmission modes, optimizing data transmission intervals, and using energy-efficient hardware components can enhance battery life.

By continuously monitoring and optimizing these key metrics, the performance and reliability of LoRaWAN-based water metering systems can be significantly improved, ensuring accurate data collection and efficient network operation.

2.4 Challenges and Solutions in LoRaWAN Implementations

2.4.1 Signal Quality and Interference

Signal quality is a critical factor in the performance and reliability of LoRaWAN networks, particularly in applications such as smart water metering where data integrity and transmission efficiency are paramount. Signal quality in LoRaWAN networks is primarily measured using two key metrics: Received Signal Strength Indicator (RSSI) and Signal-to-Noise Ratio (SNR). This section discusses the factors affecting signal quality, the impact of interference, and techniques to mitigate these issues to enhance network performance.

Factors Affecting Signal Quality

1. Distance Between Devices

The distance between the end device (e.g., water meter) and the gateway significantly impacts signal quality. As the distance increases, the RSSI typically decreases, indicating weaker signal strength. LoRaWAN is designed to support long-range communication, but optimal performance is achieved at moderate distances.

2. Physical Obstacles

Physical obstacles such as buildings, trees, and walls can obstruct the signal path, leading to attenuation and reduced signal quality. Urban environments, with their dense infrastructure, pose more significant challenges compared to open rural areas.

3. Environmental Conditions

Environmental factors such as weather conditions (rain, fog, humidity) and terrain can affect signal propagation. For instance, high humidity and heavy rain can absorb radio signals, leading to higher signal attenuation.

4. Interference from Other Devices

The presence of other electronic devices operating in the same frequency band (e.g., 868 MHz in Europe) can cause interference. This is particularly relevant in urban areas where numerous wireless devices coexist, potentially leading to signal degradation and packet loss. The primary sources of interference include:

(a) Co-channel Interference

Occurs when multiple LoRaWAN devices transmit on the same frequency channel simultaneously. This can lead to collisions and packet loss, reducing the overall network throughput.

(b) Adjacent Channel Interference

Caused by transmissions on adjacent frequency channels. While LoRaWAN's robust modulation scheme provides some resilience, excessive adjacent channel interference can still impact signal quality.

(c) Environmental Noise

Background noise from natural and man-made sources can contribute to interference. High noise levels result in lower SNR values, making it more challenging to distinguish the desired signal from the noise.

Techniques to Mitigate Interference

1) Optimal Placement of Gateways

Strategically placing gateways to minimize obstructions and maximize line-of-sight communication can significantly improve signal quality. In urban areas, deploying multiple gateways can help ensure better coverage and reduce dead zones.

2) Frequency Planning

Implementing a well-designed frequency plan that minimizes co-channel and adjacent channel interference is crucial. This includes using different frequencies for different devices and channels to avoid overlap.

3) Adaptive Data Rate (ADR)

Utilizing ADR to dynamically adjust the data rate and transmission power based on the network conditions and signal quality. ADR helps in optimizing the network performance by selecting the best data rate for each device, balancing the trade-off between range and data rate.

4) Spreading Factor Optimization

Adjusting the spreading factor (SF) for each device based on its distance from the gateway and the required data rate. Higher spreading factors increase the range but reduce the data rate, while lower spreading factors provide higher data rates at shorter distances.

5) Use of Repeaters or Relays

In environments with severe obstructions, deploying repeaters or relay nodes can help extend the network coverage and improve signal quality by retransmitting the signal to the gateway.

6) Environmental Considerations

Taking environmental factors into account during network planning and deployment. This includes avoiding installation in areas with high humidity or heavy rain and considering the terrain and building materials that may affect signal propagation.

2.4.2 Adaptive Data Rate (ADR)

Adaptive Data Rate (ADR) is a mechanism used in LoRaWAN networks to optimize the data rate, transmission power, and airtime of devices dynamically. The primary goal of ADR is to improve the network performance and battery efficiency of end devices. It achieves this by adjusting the data rate based on the radio conditions and the device's communication needs.

In a LoRaWAN network, ADR operates by allowing the network server to control the data rate and transmission power of the end devices. This control is based on the quality of the received signal, typically measured by metrics such as the Received Signal Strength Indicator (RSSI) and the Signal-to-Noise Ratio (SNR). The server periodically assesses these metrics and sends commands to the end devices to adjust their transmission parameters accordingly.

Key components of ADR

- Data Rate: The speed at which data is transmitted. Higher data rates can reduce airtime but may require better signal conditions.
- Transmission Power: The power level at which the device transmits signals. Lower power levels can conserve battery life but may reduce communication range.
- Airtime: The duration the device occupies the channel. Minimizing airtime is crucial for network efficiency, especially in dense deployments.

Benefits of ADR

- Enhanced Battery Life: By optimizing the transmission power, ADR helps reduce energy consumption, thereby extending the battery life of end devices.
- Improved Network Capacity: Adjusting data rates and minimizing airtime reduces channel congestion, allowing more devices to communicate effectively within the same network.
- Adaptability to Environmental Changes: ADR enables devices to adapt to varying radio conditions, ensuring reliable communication even in dynamic environments.

Challenges of ADR

- Complexity in Implementation: Setting up and managing ADR requires sophisticated algorithms and continuous monitoring of network performance metrics.
- Delay in Adaptation: The process of adjusting data rates and power levels may introduce delays, potentially affecting time-sensitive applications.
- Dependency on Network Conditions: ADR's effectiveness is highly dependent on the accuracy of the signal quality measurements and the responsiveness of the network server.

Strategies for Optimizing ADR Settings

- Accurate Signal Quality Assessment: Regularly measure RSSI and SNR to accurately reflect the current radio conditions. This data is crucial for making informed ADR adjustments.
- Dynamic Adjustment Intervals: Implement adaptive intervals for checking and adjusting data rates based on the stability of the network conditions. More frequent adjustments might be necessary in rapidly changing environments.
- Threshold Settings: Define appropriate threshold levels for RSSI and SNR to trigger ADR changes. These thresholds should be set considering the specific requirements of the application and the typical environmental conditions.
- Balancing Power and Data Rate: Find the optimal balance between transmission power and data rate to maximize both battery life and communication reliability. This involves iterative testing and fine-tuning.

2.4.3 NETWORK SCALABILITY AND MANAGEMENT

As the deployment of IoT devices continues to grow, the ability of a network to handle a large number of devices—referred to as network scalability—becomes a critical consideration. LoRaWAN networks, while advantageous in many aspects, face several scalability challenges:

a. Gateway Capacity: Each gateway in a LoRaWAN network can handle a finite number of simultaneous connections. As the number of devices increases, gateways can become bottlenecks, leading to potential data loss or delays in communication.
b. Channel Congestion: LoRaWAN networks operate on unlicensed frequency bands, making them susceptible to interference from other devices and networks operating on the same frequencies. High device density can exacerbate channel congestion, affecting the quality and reliability of communication.
c. Data Rate Limitations: LoRaWAN employs Adaptive Data Rate (ADR) to optimize data rates and energy consumption. However, in dense networks, managing ADR efficiently to prevent collision and ensure fair resource allocation can be challenging.
d. Downlink Constraints: The ratio of uplink to downlink messages in LoRaWAN is typically high, with fewer downlink opportunities. As the network scales, ensuring timely and reliable downlink communication, especially for critical control messages, can be difficult.

To address the scalability challenges in LoRaWAN networks, several strategies and best practices can be implemented:

a. Optimizing Gateway Placement: Strategic placement of gateways can enhance coverage and reduce the likelihood of congestion. By conducting thorough site surveys and utilizing propagation models, optimal locations for gateways can be identified to maximize network performance.
b. Frequency Diversity: Utilizing multiple frequency channels and sub-bands can alleviate congestion. LoRaWAN supports multi-channel configurations, which can distribute the load across different frequencies, reducing the chances of collision and improving overall network capacity.
c. Dynamic Channel Allocation: Implementing dynamic channel allocation mechanisms can help in efficiently managing the available spectrum. By dynamically assigning channels based on real-time network conditions, interference and congestion can be minimized.
d. Advanced ADR Algorithms: Employing sophisticated ADR algorithms that consider the network's density and traffic patterns can enhance scalability. These algorithms can dynamically adjust data rates and transmission power to optimize network performance and ensure fair resource allocation.
e. Load Balancing: Distributing the traffic load evenly across multiple gateways can prevent any single gateway from becoming a bottleneck. Load balancing techniques can be implemented at the network server level to manage traffic distribution effectively.
f. Network Slicing: Network slicing involves dividing the network into multiple virtual segments, each tailored to specific application requirements or device types. This can help in managing diverse IoT applications within the same network infrastructure, ensuring that critical applications receive the necessary resources and quality of service.
g. Edge Processing: Offloading some processing tasks to edge devices or gateways can reduce the burden on the central network server and improve response times. Edge processing can handle local data aggregation, filtering, and preliminary analysis, conserving bandwidth and enhancing network efficiency.
h. Regular Network Monitoring and Maintenance: Continuous monitoring of network performance and regular maintenance can identify and address issues proactively. Implementing network management tools and platforms can provide real-time insights into network health, device status, and performance metrics.

2.5 Security and Data Encryption in LoRaWAN

The security of LoRaWAN (Long Range Wide Area Network) is paramount due to its widespread application in the Internet of Things (IoT), where devices are often resource-constrained and deployed in potentially vulnerable environments. LoRaWAN incorporates various security mechanisms to ensure the authenticity, integrity, and confidentiality of data transmitted over the network. This section provides an overview of these mechanisms, with a focus on their implementation and effectiveness.

2.5.1 Security Mechanisms in LoRaWAN

LoRaWAN employs a multi-layered approach to security, incorporating several mechanisms to protect data and ensure secure communication. These mechanisms include encryption, authentication, and integrity checks, which work together to safeguard the network against a range of threats. LoRaWAN's security framework is based on the use of symmetric encryption, with AES (Advanced Encryption Standard) being the primary encryption algorithm. This ensures that data remains confidential as it travels between devices and the network server.

Key security features in LoRaWAN

i. *Root Keys (AppKey):* Each LoRaWAN device is provisioned with a unique root key (AppKey) during manufacturing or installation. This root key is used to generate session keys that secure communications.

ii. *Network Session Key (NwkSKey):* The NwkSKey is derived from the root key and is used to encrypt and verify the integrity of messages exchanged between the device and the network server. This key ensures that the network can authenticate the device and verify that messages have not been tampered with.

iii. *Application Session Key (AppSKey):* The AppSKey is also derived from the root key and is used to encrypt the payload of messages sent between the device and the application server. This ensures that the content of the messages remains confidential and is accessible only to the intended application.

iv. *Authentication*: Authentication in LoRaWAN is crucial for ensuring that only legitimate devices can join and communicate over the network. LoRaWAN uses two primary methods for device authentication:

- Over-the-Air Activation (OTAA): In OTAA, devices join the network by performing a join procedure, during which a device sends a join request to the network server. The server then responds with a join accept message, which includes security keys (NwkSKey and AppSKey) generated using the device's AppKey. This method provides a higher level of security as it generates fresh session keys for each session.
- Activation by Personalization (ABP): In ABP, devices are pre-provisioned with the necessary security keys (NwkSKey and AppSKey) before deployment. While ABP simplifies the deployment process, it is less secure than OTAA as it does not generate new keys for each session, making it more vulnerable to certain types of attacks.

v. *Integrity*: Integrity checks in LoRaWAN ensure that the data received by the network server has not been altered during transmission. This is achieved through the use of

Message Integrity Codes (MICs), which are appended to each message. The MIC is generated using the NwkSKey and covers the entire message, including the header and payload. When a message is received, the network server verifies the MIC to confirm the message's integrity. If the MIC does not match, the message is discarded, ensuring that tampered or corrupted messages are not processed.

2.5.2 Data Encryption Techniques in LoRaWAN

Data encryption is a critical component of LoRaWAN's security framework, ensuring the confidentiality and integrity of data transmitted across the network. LoRaWAN employs robust encryption techniques to protect data from unauthorized access and tampering, leveraging symmetric encryption algorithms and comprehensive key management practices. This section explores the primary data encryption techniques used in LoRaWAN, including symmetric encryption, key management, and end-to-end encryption solutions.

Symmetric Encryption

The cornerstone of LoRaWAN's encryption strategy is symmetric encryption, specifically utilizing the Advanced Encryption Standard (AES). AES is widely recognized for its security and efficiency, making it well-suited for the resource-constrained devices typical in LoRaWAN networks. In LoRaWAN, AES-128 is used, providing a balance between security and computational efficiency. This involves a shared secret key that both the sender (end device) and receiver (network or application server) use to encrypt and decrypt messages. The use of a 128-bit key provides a strong level of security against brute-force attacks while maintaining low power consumption, which is crucial for IoT devices with limited resources.

Key Management

Effective key management is essential for maintaining the security of encrypted communications in LoRaWAN. The process involves the generation, distribution, and renewal of cryptographic keys, ensuring that only authorized devices can access the network and communicate securely.

i. Key Generation: Keys are generated using a secure process that ensures their unpredictability and robustness. Each device is provisioned with a unique root key (AppKey) that is used to derive session keys (NwkSKey and AppSKey). The generation of these keys is typically handled by the device manufacturer or network operator, following stringent security protocols.
ii. Key Distribution: In the Over-the-Air Activation (OTAA) process, session keys are securely distributed during the join procedure. When a device sends a join request, the network server generates the session keys and transmits them back to the device in an encrypted join accept message. This dynamic distribution process enhances security by ensuring that new session keys are used for each session.
iii. Key Renewal: To maintain long-term security, session keys should be periodically renewed. This can be achieved through rejoining the network using OTAA, which triggers the generation of new session keys. Regular key renewal reduces the risk of key compromise and ensures ongoing protection against evolving security threats.

End-to-End Encryption

End-to-end encryption (E2EE) is a critical technique for ensuring that data remains confidential throughout its entire journey from the source device to the final destination application server. In LoRaWAN, E2EE can be implemented to provide an additional layer of security beyond the standard network-layer encryption. This involves encrypting the data payload at the source device using a key that is shared only with the application server, ensuring that even if the data is intercepted or accessed by intermediate nodes (such as network servers or gateways), it remains unreadable. The application server then decrypts the data payload using the corresponding decryption key. E2EE enhances data confidentiality and protection against insider threats. It ensures that sensitive data, such as personal information or critical sensor readings, is only accessible to authorized end-users and applications, mitigating the risk of data breaches.

2.5.3 Vulnerabilities and Threats

Despite the robust security mechanisms and encryption techniques implemented in LoRaWAN, the network is not immune to vulnerabilities and threats. Understanding these weaknesses is crucial for developing more secure systems and mitigating potential risks. This section explores common vulnerabilities in LoRaWAN, provides examples of security breaches, and discusses the implications of these threats on the overall security of LoRaWAN networks.

Common Vulnerabilities

LoRaWAN networks face several common vulnerabilities due to the inherent characteristics of IoT devices and the nature of wireless communication. These vulnerabilities can be exploited by attackers to compromise the network's security.

i. Replay Attacks: In a replay attack, an attacker intercepts a legitimate data transmission and retransmits it to the network. Without adequate countermeasures, such as unique message identifiers or timestamps, the network may process these retransmitted messages as if they were valid, leading to potential disruptions or unauthorized actions.
ii. Eavesdropping: Eavesdropping occurs when an attacker intercepts data being transmitted over the air. While the use of AES encryption helps protect the confidentiality of data, weaknesses in key management or insecure initial key distribution can render the encryption less effective, allowing attackers to decipher the intercepted data.
iii. Key Compromise: If the root key (AppKey) or session keys (NwkSKey and AppSKey) are compromised, attackers can decrypt data transmissions, inject false data, or impersonate legitimate devices. Key compromises can occur due to poor key management practices, inadequate physical security of devices, or sophisticated cyber-attacks.
iv. Device Cloning: Device cloning involves duplicating the credentials and configuration of a legitimate device to create a counterfeit device. This cloned device can then be used to join the network and perform malicious activities, such as sending false data or consuming network resources.

Implications of Vulnerabilities

The presence of vulnerabilities in LoRaWAN networks has several implications for the security and reliability of IoT applications:

i. Data Integrity and Confidentiality: Compromised data integrity and confidentiality can lead to misinformation, unauthorized access to sensitive information, and privacy breaches. This is particularly concerning for applications involving personal data or critical infrastructure.
ii. Network Availability: Attacks that exploit vulnerabilities can disrupt network availability, leading to denial of service (DoS) conditions. This can impact the reliability of IoT services and result in financial losses or operational disruptions.
iii. Trust and Adoption: Security vulnerabilities can undermine trust in LoRaWAN networks and hinder their adoption. Ensuring robust security is essential for gaining user confidence and encouraging the widespread deployment of LoRaWAN-based solutions.

The exploration of security and data encryption in LoRaWAN reveals a multi-faceted approach to safeguarding IoT networks. LoRaWAN's security framework is built on a foundation of robust encryption and comprehensive key management practices, which are essential for maintaining the confidentiality, integrity, and authenticity of data transmitted across the network. While LoRaWAN implements strong security mechanisms, addressing the existing vulnerabilities and evolving threats requires ongoing research and the development of advanced countermeasures. Ensuring the security and reliability of LoRaWAN networks is essential for fostering trust and promoting the widespread adoption of IoT solutions.

Conclusions

in conclusion, LoRaWAN presents a compelling solution for IoT applications, particularly for remote water consumption monitoring systems, due to its unique advantages over other Low Power Wide Area Networks (LPWANs). One of the primary strengths of LoRaWAN is its long-range communication capability, which can effectively cover extensive areas, making it ideal for widespread deployment in both urban and rural environments. This is particularly beneficial for remote water metering, where devices may be dispersed over large geographical areas and need reliable connectivity.

LoRaWAN's adaptive data rate (ADR) mechanism optimizes network performance by dynamically adjusting the data rate and transmission power, which is crucial for maintaining efficient communication and conserving the battery life of end devices. This feature ensures that water meters can operate for extended periods without frequent battery replacements, thus reducing maintenance costs and efforts. The network scalability of LoRaWAN, supported by strategies such as optimal gateway placement, frequency diversity, and advanced ADR algorithms, allows it to handle a large number of devices simultaneously. This scalability is crucial for water utilities aiming to monitor thousands of water meters across a city or region, ensuring reliable data collection and network efficiency. Moreover, LoRaWAN's capability to mitigate interference and maintain signal quality through techniques like spreading factor optimization and the use of repeaters or relays further enhances its reliability in diverse environmental conditions. This makes it suitable for deployment in areas with physical obstructions or challenging weather conditions, ensuring consistent performance.

In summary, the long-range communication, energy efficiency, robust security, scalability, and interference mitigation features make LoRaWAN an optimal choice for remote water consumption monitoring systems.

3. System Design & Implementation

3.0 Introduction

This chapter provides a detailed account of the components and architecture used to develop the LoRaWAN-based water meter monitoring system. This chapter is essential for understanding the technical foundations of the project, as it lays out the hardware and software elements, their configuration, and the overall system architecture. The design phase encompasses the selection and integration of the Intelis wSource water meter and the LoRaWAN MikroTik gateway, both critical for ensuring reliable data transmission and reception. Additionally, the chapter delves into the software aspects, including the deployment of the Chirpstack platform (LoRaWAN Network Server), the deployment of the database InfluxDB and the Grafana visualization tool, and their integration with ChirpStack. We will also go through the decryption logic and eventually provide a high-level overview of the system as a whole.

3.1 Hardware Components

3.1.1 The Intelis wSource water meter

The Intelis wSource water meter is a sophisticated smart metering device designed to deliver precise and dependable water usage data. This section provides a detailed examination of its features, technical specifications, and operational capabilities, emphasizing its suitability for integration into IoT networks, particularly LoRaWAN.

Key Features

- High Accuracy: The Intelis wSource water meter boasts high-precision sensors that ensure accurate measurement of water flow and consumption. This accuracy is vital for ensuring fair billing and effective water resource management.
- Compatibility: The Intelis wSource is available with various communication options, including wM-Bus, LoRa, Sigfox, and OMS 868MHz. It is compatible with harsh installation conditions and supports multiple data collection systems, ensuring flexibility and scalability for utilities.
- Long Battery Life: The meter is designed with energy efficiency in mind. It operates on a long-lasting lithium battery, which can sustain the device for several years, depending on the usage and reporting frequency. This feature minimizes maintenance efforts and costs associated with battery replacement.
- Robust Construction: The Intelis wSource water meter is built to endure harsh environmental conditions. It has a durable housing that protects it from water ingress, dust, and physical impacts. The device is rated IP68, indicating its resilience to submersion in water and complete protection against dust.

- Tamper Detection: The meter includes tamper detection features to alert operators of any unauthorized attempts to interfere with the device. This functionality is crucial for maintaining data integrity and preventing fraudulent activities.
- User-Friendly Interface: The water meter has a user-friendly interface that allows for easy installation and configuration. The interface provides visual indicators for status monitoring, which aids technicians during setup and maintenance.
- Interoperability and Scalability: The Intelis wSource is interoperable with open standards and non-proprietary communication protocols, allowing seamless integration with various data management systems. It supports both Automatic Meter Reading (AMR) and Advanced Metering Infrastructure (AMI) technologies, ensuring scalability and flexibility for utilities.

Technical Specifications.

- Measurement Range: The Intelis wSource is the only water meter on the market to be MID-certified for R1000 accuracy, a new standard in precision for measuring water consumption at low flow rates. It features a low-to-no maintenance solid-state ultrasonic technology that maintains accuracy in harsh conditions and delivers long-lasting performance in the field.
- Operating Temperature: The Intelis wSource is designed to function in a wide range of temperatures, typically from -10°C to 55°C, ensuring reliability in various climates.
- Pressure Rating: The device can withstand high water pressures, with a typical pressure rating of up to 16 bar, making it suitable for high-pressure water systems.
- Communication Interface: The integrated LoRaWAN module supports Class A and Class C devices, providing flexibility in communication patterns. It operates within the 863-870 MHz frequency band for Europe, ensuring compatibility with regional networks.
- Durability and Installation: The meter is IP68-rated for waterproofing and has a battery life of up to 22 years, making it suitable for installation in flooded manholes.

Operational Capabilities

- Data Collection and Transmission: The Intelis wSource water meter continuously collects water usage data. At predefined intervals, it transmits this data to the LoRaWAN gateway. The transmission intervals can be configured based on the monitoring requirements, ranging from a few minutes to several hours.
- Remote Configuration and Updates: The meter supports remote configuration and firmware updates over the air (OTA). This capability allows operators to adjust settings and deploy updates without needing physical access to the device, enhancing operational efficiency.
- Data Security: Security is a critical aspect of the Intelis wSource water meter. The device uses advanced encryption standards to protect data during transmission, ensuring that the information remains secure from unauthorized access.
- Integration with IoT Platforms: The Intelis wSource is designed for seamless integration with various IoT platforms, including Chirpstack. This integration enables comprehensive data analysis, visualization, and reporting, facilitating informed decision-making for water management.

In conclusion, the Intelis wSource water meter is a pivotal component in the LoRaWAN-based remote water meter monitoring system. Its high accuracy, robust construction, and advanced communication capabilities make it well-suited for reliable and efficient water usage monitoring. Understanding the features and technical specifications of the Intelis wSource water meter is essential for optimizing its performance within the IoT network and ensuring accurate and timely data collection for water resource management.

Fig.4The Intelis wSource water meter

3.1.2 THE MIKROTIK WAP LR8 GATEWAY

The MikroTik wAP LR8 kit is an out-of-the-box solution designed to use LoRaWAN technology. This kit includes a pre-installed UDP packet forwarder to any public or private LoRa servers and an outdoor weatherproof wireless access point with a 2.4 GHz WLAN interface and Ethernet port with a maximum data rate of 300 Mbit/s. The wAP LR8 kit supports the 863-870 MHz frequency, commonly used in the European Union and it is positioned to maximize coverage and ensure reliable data transmission from the water meters deployed within its range. Key features of this gateway include:

Key Features

Multi-Channel Support: The gateway supports multiple frequency channels, allowing it to handle data from several devices simultaneously.

High Sensitivity: Equipped with high-sensitivity receivers, the gateway can capture weak signals from distant end devices, ensuring reliable data transmission even in challenging environments.

Network Compatibility: The gateway is compatible with the Things Stack platform, facilitating seamless integration and management of the LoRaWAN network.

Scalability: The gateway can support a large number of end devices, making it suitable for scalable IoT deployments.

Weatherproof Design: The wAP LR8 is housed in a durable enclosure, making it suitable for outdoor installations where it may be exposed to harsh environmental conditions.

LoRaWAN Gateway: This device supports LoRaWAN, a low-power, wide-area networking protocol designed to wirelessly connect battery-operated devices to the internet.

Wireless Access Point: It functions as a wireless access point, providing standard Wi-Fi connectivity in addition to LoRa capabilities.

High-Gain Antenna: Equipped with a high-gain antenna, the wAP LR8 ensures extended range and reliable communication with IoT devices.

Power Options: The device supports multiple power options, including PoE (Power over Ethernet), making it versatile and easy to deploy in various environments.

Technical Specifications

Processor: Qualcomm Atheros QCA9531 650 MHz.

Memory: 64 MB RAM, 16 MB flash storage.

Antenna Gain: 2.5 dBi for 2.4 GHz, 3.5 dBi for LoRa.

Power Consumption: Max 7 W.

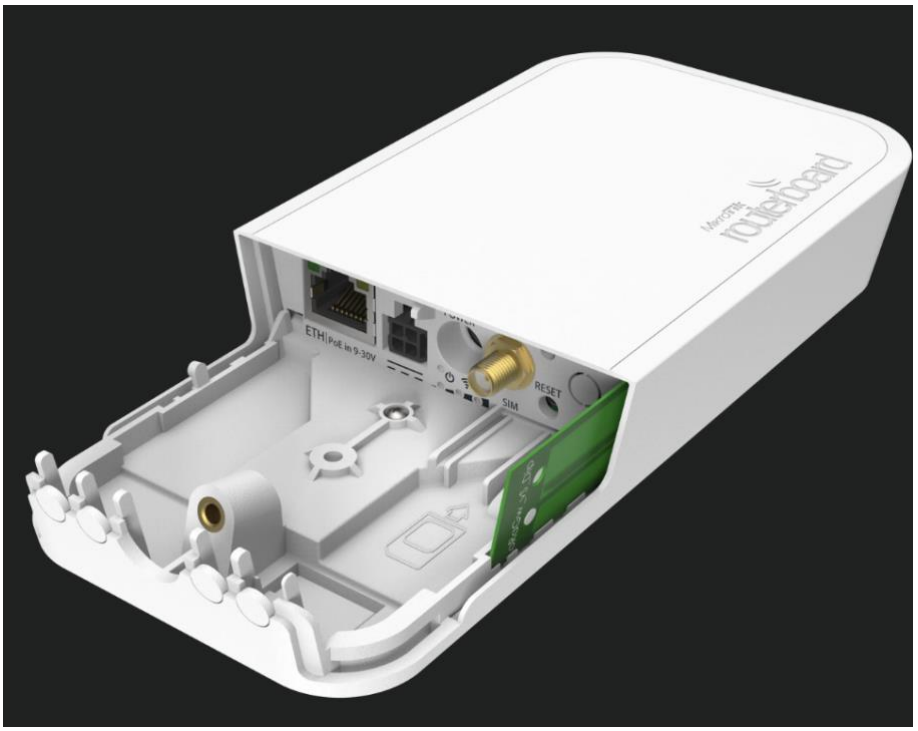


Fig.5 The MikroTik wAP LR8

3.2 SOFTWARE COMPONENTS

In this section, we describe the key technologies utilized in the development and deployment of the LoRaWAN network server, data processing, storage, and visualization components. The main technologies used in this project are ChirpStack, Flask, InfluxDB, and Grafana.

3.2.1 CHIRPSTACK NETWORK SERVER

ChirpStack is an open-source LoRaWAN network server stack. It provides the necessary components for building and managing a LoRaWAN network, including device management, network server, and application server functionalities.

Features

- LoRaWAN Specification Compliance: ChirpStack adheres to the LoRaWAN specifications, ensuring compatibility with various LoRaWAN devices.
- Scalability: Designed to scale horizontally, making it suitable for both small and large deployments.
- Flexibility: Offers configurable integrations for device management and application layer data forwarding.

ChirpStack was used to manage the LoRaWAN sensors, handle uplink communications, and forward data to the Flask application via webhooks. Its role was critical in ensuring reliable data transmission from the sensors to the server.

3.2.2 FLASK

Flask is a lightweight WSGI web application framework in Python. It is designed with simplicity and flexibility in mind, allowing developers to build scalable web applications quickly.

Features

- Microframework: Minimalistic design with no mandatory dependencies on external libraries or components.
- Extensibility: Supports extensions for adding functionalities such as authentication, database interaction, and more.
- Ease of Use: Simple and easy to set up, making it ideal for building RESTful APIs and handling webhooks.

Flask was used to create a web application that receives data from the ChirpStack network server. The application decrypts the payload, processes it, and stores the relevant information in InfluxDB. Flask's ability to handle web requests and its compatibility with Python's extensive library ecosystem made it an excellent choice for this project.

3.2.3 INFLUXDB

InfluxDB is an open-source time series database developed by InfluxData. It is optimized for handling high write and query loads, making it ideal for storing time-stamped data such as IoT sensor readings.

Features

- Time Series Optimized: Efficiently stores and queries time series data.
- High Performance: Capable of handling high write and query throughput.
- Flux Query Language: Advanced querying capabilities with the Flux scripting language.
- Retention Policies: Allows automatic data retention and downsampling.

InfluxDB was used to store the decrypted sensor data. Its time-series optimization ensured efficient storage and retrieval of time-stamped data, which is crucial for monitoring and analyzing sensor readings over time. InfluxDB's integration with Grafana enabled seamless visualization of the stored data.

3.2.4 GRAFANA

Grafana is an open-source analytics and interactive visualization web application. It allows users to query, visualize, and understand their data, regardless of where it is stored.

Features

- Data Source Flexibility: Supports multiple data sources, including InfluxDB, Prometheus, Elasticsearch, and more.
- Interactive Dashboards: Provides customizable and interactive dashboards for real-time data visualization.
- Alerting: Offers alerting functionalities to notify users of critical events based on data thresholds.
- Templating: Allows the creation of reusable dashboard templates for consistent visualization across different datasets.

Grafana was used to create dashboards for visualizing the sensor data stored in InfluxDB. Its ability to integrate with InfluxDB and provide real-time, interactive visualizations made it an essential tool for monitoring and analyzing sensor performance. Grafana's dashboard and alerting capabilities ensured that users could easily track and respond to changes in sensor readings.

The combination of ChirpStack, Flask, InfluxDB, and Grafana provided a robust and scalable solution for managing, processing, storing, and visualizing IoT sensor data in a LoRaWAN network. Each technology played a crucial role in the system architecture, contributing to the overall functionality and success of the project.

3.3 System Architecture

The system architecture of this project is designed to provide a robust and scalable solution for managing and visualizing sensor data in a private LoRaWAN network. The architecture integrates multiple technologies, including ChirpStack for managing LoRaWAN communication, a Flask application for data processing, InfluxDB for time-series data storage, and Grafana for data visualization. This section provides a detailed overview of the system components and their interactions.

3.3.1 System Overview

The system architecture of this project is designed to provide a comprehensive solution for monitoring and visualizing water consumption data using a private LoRaWAN network. The system architecture can be divided into three main layers:

1. Data Collection Layer

- Intelis wSource (End device): The Intelis wSource water meters are deployed to collect data such as water consumption. These devices send data packets wirelessly to the LoRaWAN gateway. Each water meter is equipped with a unique identifier (devEUI) and an encryption mechanism that encrypts the payload before sending it to the gateway to ensure data security.
- MikroTik Gateway: The gateway receives the encrypted data packets from the Intelis wSource meters and forwards them to the ChirpStack Network Server over the internet.

2. Data Processing Layer

- ChirpStack Network Server: It handles the LoRaWAN protocol, manages the network of LoRaWAN devices, and ensures reliable communication between the end devices and the application server. The server also decodes the preamble LoRaWAN frame and forwards the rest of the payload which remains encrypted to the Flask application via a webhook.
- Flask Application: The Flask application serves as the middleware between ChirpStack and InfluxDB. Its responsibilities include:
  - Receiving the encrypted payload data from ChirpStack by listening to requests.
  - Using predefined AES keys to decrypt the payload data received.
  - Processing the decrypted data to extract relevant information.
  - Storing the processed data in a text file for backup.
  - Preparing the data for storage in InfluxDB.

3. Data Storage and Visualization Layer

- InfluxDB: The Flask application writes the processed sensor data into InfluxDB. Each data point includes:
  - Measurement unit (e.g., liters).
  - Total consumption.
  - Interval (e.g., hourly).
  - Hourly consumption values.
- Grafana: Grafana connects to InfluxDB and retrieves the stored sensor data to create real-time dashboards customed to our specific needs.

Fig. 6 Data flow of the process

3.3.2 Steps for System Setup

The system setup section outlines the steps taken to deploy and configure the various components of the IoT solution, including the ChirpStack network server, Flask application, InfluxDB, and Grafana. Each component plays a critical role in ensuring seamless data collection, processing, and visualization.

The order followed for setting up the system was:

1) Deploying Chirpstack locally on the servers of TUC. Chirpstack provides comprehensive documentation on how to do that: https://www.chirpstack.io/docs/getting-started/debian-ubuntu.html
2) Configuring the gateway to send the received data packets to Chirpstack: https://www.thethingsindustries.com/docs/gateways/models/mikrotikrouterboard/
3) Defining a payload codec on Chirpstack to decode the non-encrypted first byte of the received payload into human-readable format.
4) Setting up an HTTP integration that forwards events from Chirpstack to a user-configurable endpoint as POST requests.
5) Setting up a Flask application to listen to those post requests.
6) Write code to handle payload decryption and store data in InfluxDB.
7) Integrate InfluxDB with Grafana to create custom real-time dashboards to monitor water consumption.

The steps (1) and (2) were easy to accomplish just by following the orders from the attached documentation. In the next and last two sections of this chapter, we will go through steps (3), (4), (5), (6) and (7).

3.4 DATA PROCESSING

This section describes the mechanisms that were developed to decode the data transmitted by Intelis wSource water meters. The process includes understanding the structure of the data objects (DOBJ) that the water meters send, developing a decryption mechanism, storing the decrypted data on a file locally, developing a decoder to parse the decrypted payloads, and interpreting the information to meaningful values.

The steps taken for this process were:

1) Analyzed the LoRaWAN frame and extracted useful frame info from the non-encrypted first three bytes.
2) Develop a decryption mechanism based on the previously extracted info to remove the sensor's Applicative Encryption and access the raw payload.
3) Develop a decoder to parse the raw payload and Interpret the DOBJ to meaningful values according to the manufacturer's datasheet.

Each of these steps is discussed separately in the following sub-sections.

3.4.1 LoRaWAN Preamble Frame Analysis

To understand the decryption and decoding logic, we should first understand how the Intelis wSource structures the data in the payload. In this section, the payload format will be discussed along with the methods used to extract the relevant information in the pre-decryption phase.

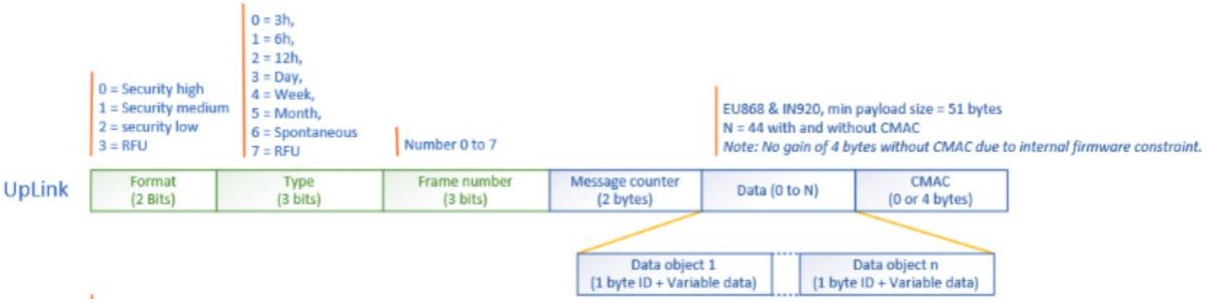


Fig.7 Uplink payload format

The first byte is known as the "preamble", as shown in the figure above is composed of Frame Format, the Type of frame that refers to the periodicity of the frame (3h, 6h, …), and the Frame Number which refers to the time of emission linked to the period. This is defined as the preamble of the frame and this byte is never encrypted.

- The first two bits will refer to the Format of the data payload, it will tell which security level is used for the frame (applicative Encryption and Authentication or not).
- The Frame Type will allow the Integrator to understand the periodicity of the frame. This periodicity will be specific to each frame and can take the following period: 3h, 6h, 12h, Daily, Weekly, or Monthly.
- The Frame Number will allow based on the periodicity we got from the Frame Type to know which frame we received and so when it has been generated.

For example, in the case of a 3h periodicity frame (Frame Type = 0), the Intelis wSource will emit 8 different frames in the day. So, frame Number 0 will refer to the frame sent between 00h and 03h, and so generated at 00h. Frame Number 1 will refer to the frame sent between 03h and 06h, and so generated at 03h, etc.

In another example with a frame periodicity of 12h (Frame Type = 2), the Intelis wSource will send 2 frames in the day. The Frame Number 0 will refer to the frame sent between 00h and 12h, and so generated at 00h. Frame Number 1 will refer to the frame sent between 12h and 00h, and so generated at 12h, and the day after we restart with frame Number 0.

The frames are sent at a random time inside the associated period for daily or sub-daily periods and sent at a random time on the first day of the week/month for weekly/monthly frames.

A Specific case will appear with a Daily frame periodicity (Frame Type = 3) as the Intelis wSource will only send 1 frame per day, the Frame Number will refer to the day of the week. In that case, Frame Number 0 will refer to Monday, Frame Number 1 to Tuesday, … until Sunday which is Frame Number 6. For the Frame periodicity of Week, Month, and spontaneously, the Frame Number will be set to 0.

	Frame Type						
	0	1	2	3	4	5	6
Frame Number	3	6	12	Daily	Weekly	Monthly	Spontaneous
0	0	0	0	Monday	N/A	N/A	N/A
1	3	6	12	Tuesday	N/A	N/A	N/A
2	6	12	N/A	Wenesday	N/A	N/A	N/A
3	9	18	N/A	Thursday	N/A	N/A	N/A
4	12	N/A	N/A	Friday	N/A	N/A	N/A
5	15	N/A	N/A	Saturday	N/A	N/A	N/A
6	18	N/A	N/A	Sunday	N/A	N/A	N/A
7	21	N/A	N/A	N/A	N/A	N/A	N/A

Fig.8 Screenshot from the documentation provided by Itron to understand Frame Type

The two next bytes are coded on LSB (Less Significant Bytes) and will refer to the Frame Message Counter. This counter will be incremented by 1 on every transmission including retries.

Then the following bytes will contain the data sent by the Intelis wSource which also contains the water consumption values. It is composed of the Intelis wSource Data Object (DOBJ).

The last 4 bytes are coded MSB First and will compose the CMAC for the authentication of the message if the message is authenticated, if not those 4 bytes are not included in the payload.

For example, let's consider the payload: 041400187D3CF6028C56FD99.

Decomposing the frame as explained above, the integrator will obtain:

- Preamble (1byte) = $04_{(16)} = 00000100_{(02)}$, which means that:

  i) Frame Format = $00_{(02)} = 00_{(10)}$ => Security level is High => Data Object is Encrypted and Authenticated.

ii) Type = $000_{(02)}$ = $00_{(10)}$ => 3h periodicity.
iii) Frame Number = $100_{(2)}$ = $4_{(10)}$ => Frame sent between 12h and 15h.

- Message Counter (2bytes, LSB first) = $0014_{(16)}$ = $20_{(10)}$ => message counter is 20.
- Data = $187D3CF602_{(16)}$, From this, we will need to remove the Applicative encryption to access the actual data sent. That process is shown in the next sub-chapter because, in reality, the Data part is 44 bytes MSB first.
- CMAC = $8C56FD99_{(16)}$. We can use the CMAC to authenticate our messages for extra security. It is always 4 bytes MSB first.

It is worth observing that the total length of each payload is 51 bytes.

We just described how we analyzed the LoRaWAN frame and extracted useful frame info from the non-encrypted first three bytes. The next sub-section is about developing a decryption mechanism based on the previously extracted info to remove the sensor's Applicative Encryption and access the raw payload.

3.4.2 Decryption Mechanism

In this sub-section, we will explain the process of decrypting the data received from Intelis wSource water meters. This decryption mechanism ensures that the data transmitted over the network is secure and can only be interpreted by authorized systems.

Components of the Decryption Process

1) Initialization Vector (IV): The IV is constructed by reading the DevEUI (LSB first) for each device separately. Then we add the message counter (LSB first) we read from the preamble of the message we want to decrypt. The rest of the string should be filled by zeros until 16 bytes are reached. For example, for "DevEUI: 000781377000022E, and hypothetical Message counter: 1400, the IV should be 2E020070378107000014000000000000".

2) Device EUI (devEUI): The devEUI is a globally unique identifier assigned to each water meter in production and provided by the manufacturer (Itron). It is an 8-byte string that ensures that the data can be traced back to a specific device.

3) Message Counter: The message counter is a sequential number that increments with each message sent by the device. It helps in maintaining the integrity and order of the messages. By using the message counter in the decryption logic, we make sure that every IV will be uniquely matched to the desired payload.

4) AES 128-bit Key: The AES (Advanced Encryption Standard) key is a 128-bit secret key used for both encryption and decryption of the data. This key is also injected in the Intelis wSource during production and it is provided by the manufacturer.

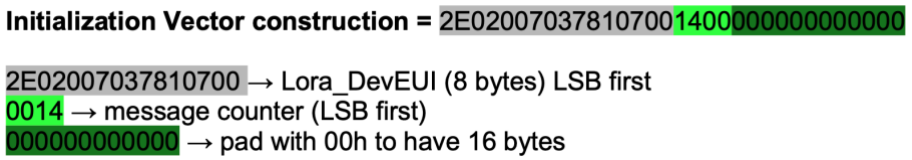


Fig.9 IV construction example from Itron's datasheet

Decryption Using AES 128-bit CTR Mode:

- The decryption is performed using the AES algorithm in Counter (CTR) mode. This mode allows the decryption of data blocks independently and is suitable for streaming data.
- The previously constructed IV and the AES 128-bit key are used to initialize the AES decryption process.
- The encrypted data is then decrypted using this initialized AES instance, converting the ciphertext back into plaintext.

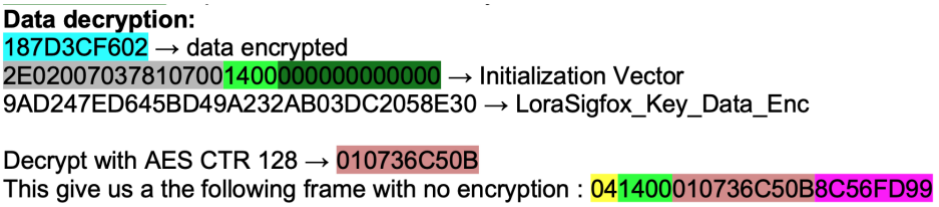


Fig.10 Decryption example from Itron's datasheet

This decrypted payload will represent the raw data sent by the device in HEX form. From that point, we still have to decode from HEX to decimal to transform the data into human-readable form. That step will be discussed in the following section, after explaining the decryption mechanism.

Upon receiving an encrypted payload from Chirpstack, the first step is to log the request and parse the incoming data to extract the devEUI, message counter, and the encrypted payload.

Explanation of the Process Flow

1. Initialization and Preparation.
2. Construct the IV.
    - The devEUI is converted to a byte array and reversed.
    - The message counter is converted to a byte array in Little Endian format.
    - These two-byte arrays are combined with six trailing zeros to form the IV.
3. Decode the Encrypted Payload.
    - The Base64 encoded payload is decoded to get the raw encrypted data.
    - The relevant part of the payload is extracted for decryption.
4. Perform Decryption.
    - The AES 128-bit key and the constructed IV are used to initialize the AES decryption process in CTR mode.
    - The encrypted data is decrypted to produce the original plaintext data.

The Decryption Process

To further understand the decryption process, let's look at the event's details of an actual uplink received on 25/6/2024 at 21:58:21, to see how our application extracts the relevant data we need for the IV construction.

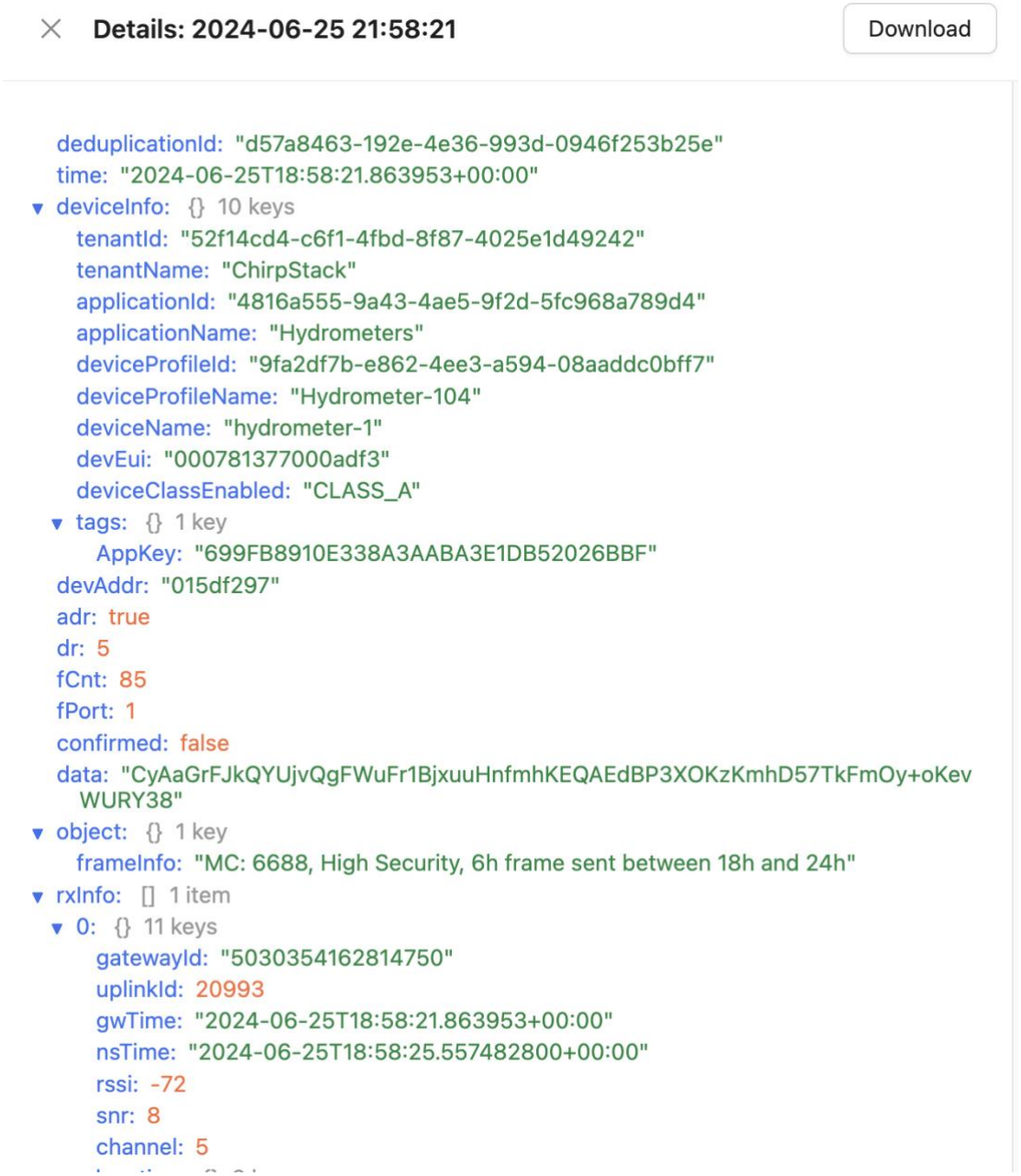


Fig.11 Uplink event details as shown in Chirpstack

We see that both details- DevEUI, and MC (which means message counter) needed for the construction of the IV are present in the event details under the tags "devEUI" and "frameInfo".

That is because, when configuring the Chirpstack network server, we created a "Device profile" for our sensors. By doing so, we were not only able to describe to Chirpstack unique information about our water meters (Fig 12), but also, define JavaScript functions inside the platform that were responsible for handling the decoding of the preamble. (Fig. 13). If the Intelis wSource would not

apply encryption in the payloads, we could configure the Payload Codec to extract the actual data sent. However, in our case, it is necessary to handle the decryption externally using Python libraries since the internal payload codec section of Chirpstack does not support the desired cryptographic libraries. In Fig.11 we also see the encrypted payload sent under the tag “data”.

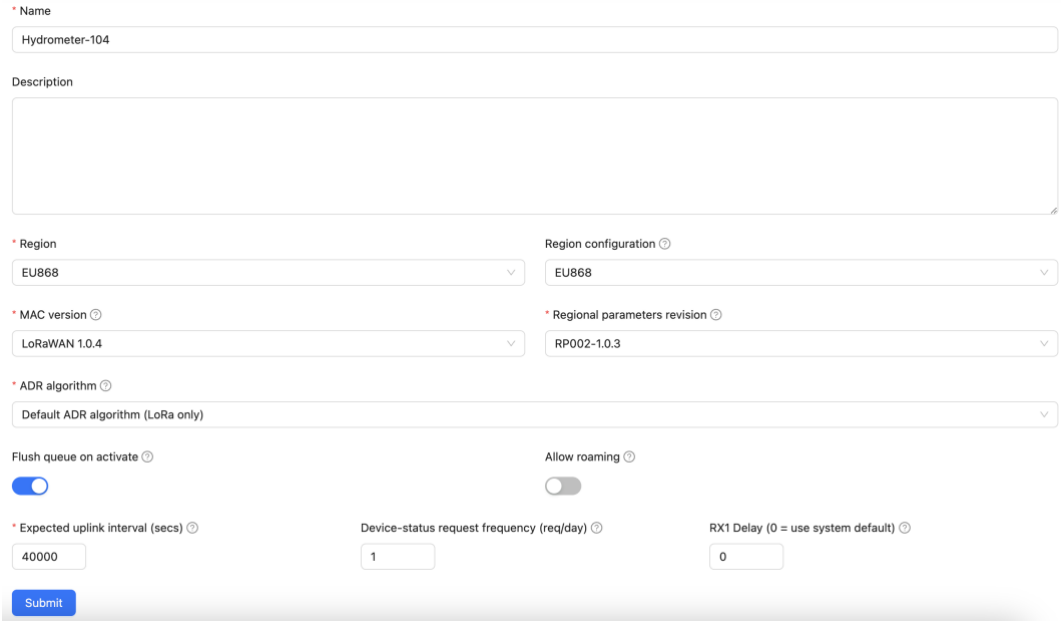


Fig.12 Defining a “device profile” in Chirpstack.

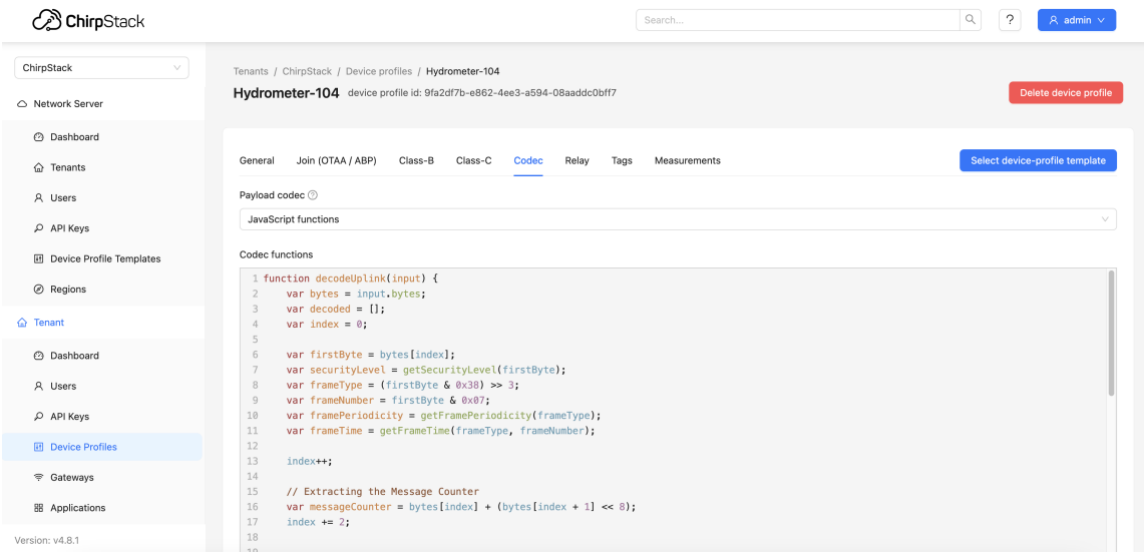


Fig. 13 Defining a payload Codec.

Now that we know both DevEUI and the message counter we are ready to build the decryption mechanism. For that, we will leverage the Chirpstack webhooks integration to forward the received data from Chirpstack to our application that handles decryption.

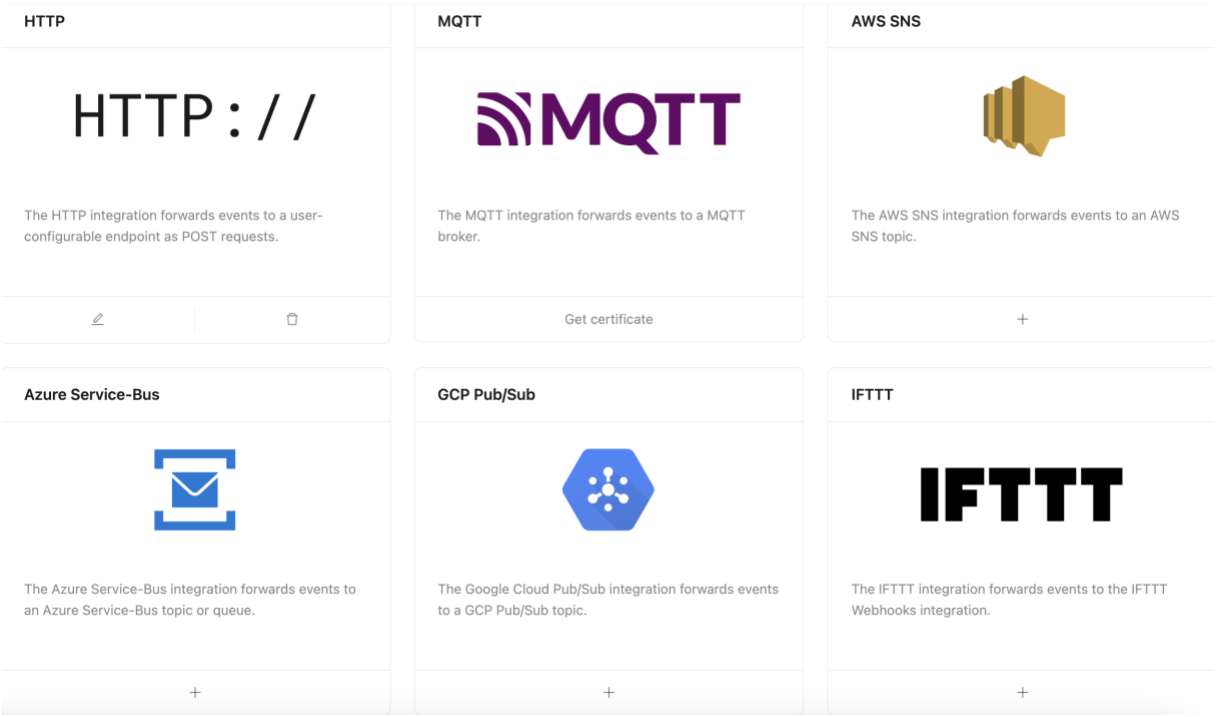


Fig.14 Selecting HTTP to set up a webhook

ChirpStack will make a POST request to this URL (Fig.15) with 'event' as a query parameter. Multiple URLs can be defined as a comma-separated list. Whitespaces will be automatically removed.

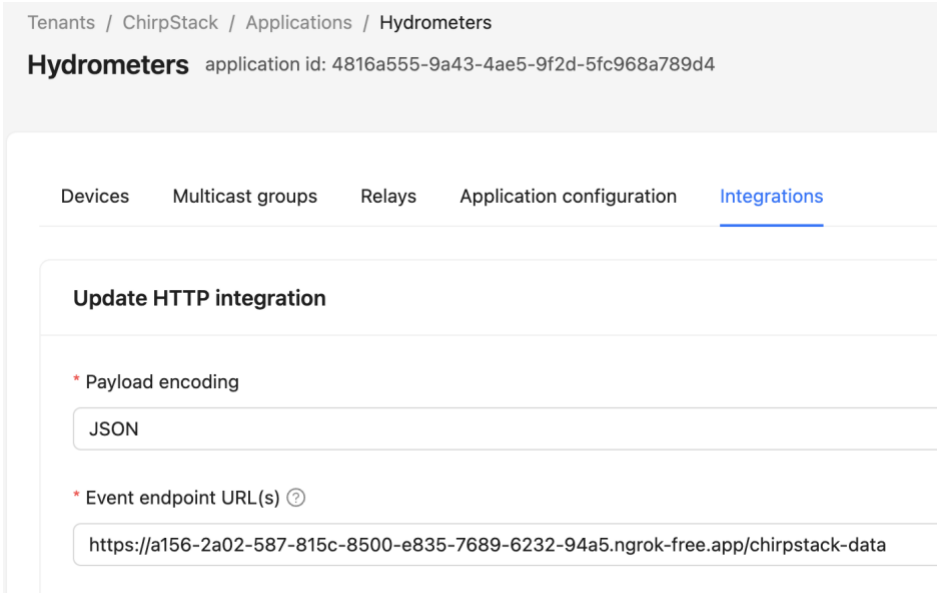


Fig.15 Defining the URL that Chirpstack will forward the data into

Because Chirpstack is running on the servers of TUC, we need another tool that will act as a tunnel for the data to be transferred to our local host. For that purpose, we used Ngrok:

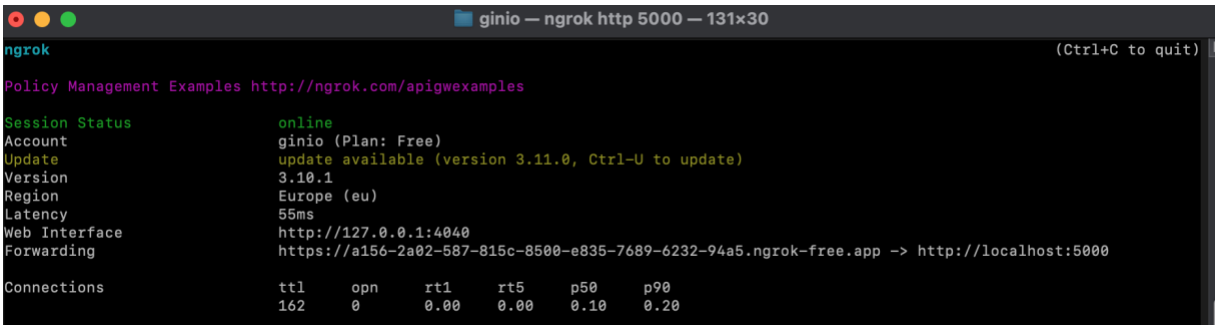


Fig. 16 Terminal instance showcasing the running Ngrok

We see in the "Forwarding" row that Ngrok listens to the event endpoint we have defined on Chirpstack and forwards the data to our local host on port 5000.

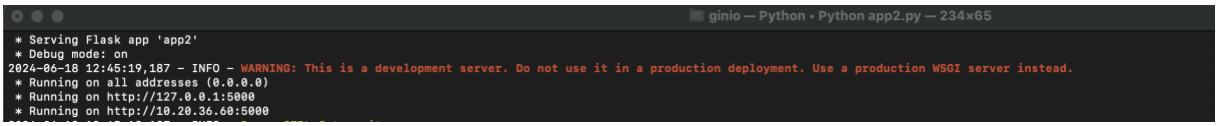


Fig. 17 Terminal instance showcasing the Flask app running on port 5000

That way we ensure that every payload received by Chirpstack, will automatically be forwarded to our local host for further processing and decryption. Thus, by Choosing a programming language and cryptography library that supports AES decryption in CTR mode the decryption process can be implemented. For that purpose, we used the PyCryptodome library and Python:

```python
def construct_iv(dev_eui, message_counter):
    if message_counter is None:
        raise ValueError("Message counter is None")
    # Convert DevEUI and message counter to byte arrays (LSB first)
    dev_eui_bytes = bytes.fromhex(dev_eui)[::-1]
    counter_bytes = message_counter.to_bytes(2, 'little')

    # Construct the IV with DevEUI, counter, and 6 trailing zeros
    iv = dev_eui_bytes + counter_bytes + b'\x00' * 6
    print(f"Constructed IV: {iv.hex()}")  # Print the IV in hexadecimal format

    return iv

from Crypto.Cipher import AES

def decrypt_data(encrypted_data, key, iv):
    cipher = AES.new(key, AES.MODE_CTR, nonce=b'', initial_value=iv)
    decrypted_data = cipher.decrypt(encrypted_data)
    print("Decryption successful")  # Confirmation of decryption
    return decrypted_data

def store_payload(decrypted_data, filename='decrypted_payload.txt'):
    with open(filename, 'wb') as file:
        file.write(decrypted_data)
    print(f"Decrypted payload stored in {filename}")  # Confirmation of storage
```

Fig. 18 Decryption logic manifested in Python code

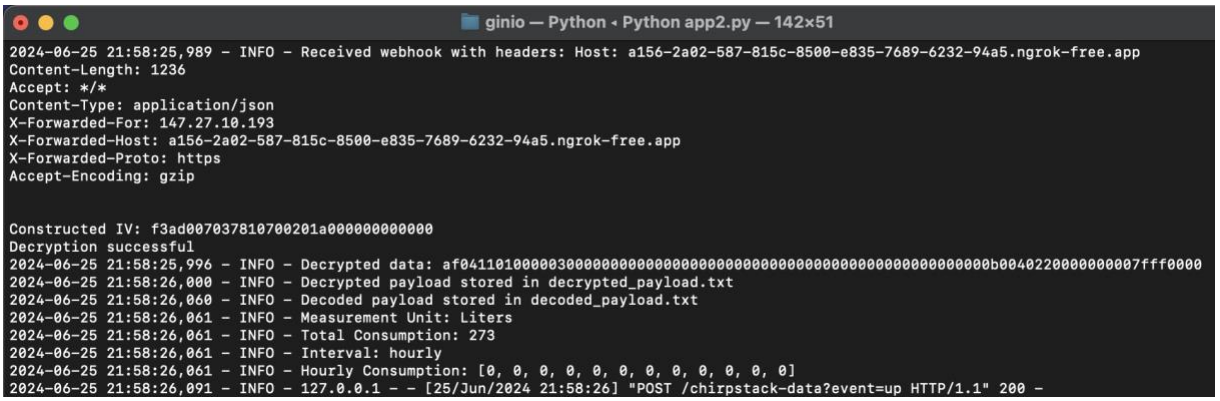


Fig. 19 Terminal instance from running the Decryption mechanism in action

As shown in Fig.19 and Fig.12, ChirpStack received at 2024-06-25 21:58:21 the payload: "CyAaGrFJkQYUjvQgFWuFr1BjxuuHnfmhKEQAEdBP3XOKzKmhD57TkFmOy+oKevWURY38"

and at 2024-06-25 21:58:25 we received at our self-host the same payload decrypted: "af04110100000300000000000000000000000000000000000000000000000000b00402200000000007fff0000".

So far, we have described two critical steps in the process: firstly, the analysis of the LoRaWAN frame, focusing on extracting useful frame information from the non-encrypted first three bytes, and secondly, the development of a decryption mechanism that utilizes this extracted information to bypass the sensor's Applicative Encryption, thereby gaining access to the raw payload. In the following chapter, we will address the third step, which involves developing a decoder to parse the raw payload and interpret the Device Object (DOBJ) into meaningful values as per the manufacturer's datasheet.

3.4.3 THE DECODER

As explained before and as also shown in Fig.20, after the decryption, the payload still needs to be decoded to transform the payload into meaningful water consumption values. For that, we developed a decoder according to the DOBJ table that the manufacturer has provided. The goal is to create a program that can read the decrypted data from decrypted_payload.txt, interpret them based on the Data Object (DOBJ) definitions and conversion rules, and then store the interpreted water consumption values for further analysis.

ID (dec)	Data Object Name	Parameters	Description	Size (bytes)	Format	Range	Unit
151	Meter Config	Unitary Value / Pulse Value	Last LCD Digit weight	1	u8, LSB First = b6..b0 = Unitary Volume b7 = Water Temperature Unit	Full	Bit 6-0 : 3: 100mL (5_4_M3); 4: 1L (6_3_M3); 5: 10L (7_2_M3); 6: 100L (8_1_M3); 7: 1m3 (9_0_M3); 50: 0,01 gallons (7_2_GAL); 51: 0,1 gallons (8_1_GAL); 101: 0,001 CuFt (6_3_CUFT); 102: 0,01 CuFt (7_2_CUFT)
		Water Temperature Unit	Water temperature Unit			Full	Bit 7: 0: CELSIUS 1: FAHRENHEIT
152	Register Digits	Digits	Number of digits	1	u8	8 or 9	Digit number
154	Meter Product Info	Product Version	See DOBJ ID 150	6	Identical DataObject as ID 150		
		Reserved		8	Reserved		
155	Battery Lifetime	Battery lifetime	Remaining Battery Life Time	1	u8	0 to 255	Month
158	Meter ID	Meter ID	Meter Serial Number	17	u8[17], LSB First	32 to 126	ASCII char table
159	Radio Product Info	Product Version	See DOBJ ID 150	6	Identical DataObject as ID 150		
		Reserved		6	Reserved		
167	Radio Stats	Battery lifetime	Remaining Battery Life Time	1	u8	0 to 255	Month
		Reserved		41	Reserved		
170	FDR 12 Delta Step 24 S24	Meter Config	See DOBJ ID 151	1	Identical DataObject as ID 151		
		Volume Database Global Index	Last record of the Global Volume Index N	4	u32, LSB First	0 to 999999999 (9 Digits) 0 to 99999999 (8 Digits) 0xFFFFFFFF = invalid value	Unitary Volume (Target turn)
		FDR Config	See DOBJ ID 4	1	Identical DataObject as ID 4		
		Interval Last12 Step24 S24	Last 12 Interval of data with step of 24 (N - N-24; N-24 - N-48; ... ; N-264 - N-288)	36	s24[0 to 11], LSB First	-8388608 to 8388605; 7FFFFEh = value in error; 7FFFFFh = no value	Unitary Volume (Target turn)
171	FDR 12 Delta Step 4 S16	Meter Config	See DOBJ ID 151	1	Identical DataObject as ID 151		
		Volume Database Global Index	Last record of the Global Volume Index N	4	u32, LSB First	0 to 999999999 (9 Digits) 0 to 99999999 (8 Digits) 0xFFFFFFFF = invalid value	Unitary Volume (Target turn)
		FDR Config	See DOBJ ID 4	1	Identical DataObject as ID 4		
		Interval Last12 Step4 S16	Last 12 Interval of data with step of 4 (N - N-4; N-4 - N-8; ... ; N-44 - N-48)	24	s16[0 to 11], LSB First	-32768 to 32765; 32766 = value in error; 32767 = no value	Unitary Volume (Target turn)
172	FDR 24 Delta S8	Meter Config	See DOBJ ID 151	1	Identical DataObject as ID 151		
		Volume Database Global Index	Last record of the Global Volume Index N	4	u32, LSB First	0 to 999999999 (9 Digits) 0 to 99999999 (8 Digits) 0xFFFFFFFF = invalid value	Unitary Volume (Target turn)
		FDR Config	See DOBJ ID 4	1	Identical DataObject as ID 4		
		Interval Last24 Step1 S8	Last 24 Interval of data with step of 1 (N - N-1; N-1 - N-2; ... ; N-23 - N-24)	24	s8[0 to 23], LSB First	-128 to 125 126 = value in error; 127 = no value	Unitary Volume (Target turn)
173	FDR 12 Delta S24	Meter Config	See DOBJ ID 151	1	Identical DataObject as ID 151		
		Volume Database Global Index	Last record of the Global Volume Index N	4	u32, LSB First	0 to 999999999 (9 Digits) 0 to 99999999 (8 Digits) 0xFFFFFFFF = invalid value	Unitary Volume (Target turn)
		FDR Config	See DOBJ ID 4	1	Identical DataObject as ID 4		
		Interval Last 12 Step 1 S24	Last 12 Interval of data with step of 1 (N - N-1; N-1 - N-2; ... ; N-11 - N-12)	36	s24[0 to 11], LSB First	-8388608 to 8388605; 7FFFFEh = value in error; 7FFFFFh = no value	Unitary Volume (Target turn)
174	FDR 2 Delta S32	Meter Config	See DOBJ ID 151	1	Identical DataObject as ID 151		
		Volume Database Global Index	Last record of the Global Volume Index N	4	u32, LSB First	0 to 999999999 (9 Digits) 0 to 99999999 (8 Digits) 0xFFFFFFFF = invalid value	Unitary Volume (Target turn)
		FDR Config	See DOBJ ID 4	1	Identical DataObject as ID 4		
		Interval Last 2 Step 1 S32	Last 2 Interval of data with step of 1 (N - N-1; N-1 - N-2)	8	s32[0 to 1], LSB First	-2147483648 to 2147483645 7FFFFFFEh = value in error; 7FFFFFFFh = no value	Unitary Volume (Target turn)
175	FDR 12 Delta S16	Meter Config	See DOBJ ID 151	1	Identical DataObject as ID 151		
		Volume Database Global Index	Last record of the Global Volume Index N	4	u32, LSB First	0 to 999999999 (9 Digits) 0 to 99999999 (8 Digits) 0xFFFFFFFF = invalid value	Unitary Volume (Target turn)
		FDR Config	See DOBJ ID 4	1	Identical DataObject as ID 4		
		Interval Last12 Step1 S16	Last 12 Interval of data with step of 1 (N - N-1; N-1 - N-2; ... ; N-11 - N-12)	24	s16[0 to 11], LSB First	-32768 to 32765; 32766 = value in error; 32767 = no value	Unitary Volume (Target turn)

Fig.20 Part of the DOBJ Table that Itron provided

As we see in the above Figure, in column "ID (dec)" we see the byte(s) that will be part of the raw payload sent by the water meter that needs (s) to be parsed to determine the DOBJ (data objects) that have been sent. In the B column "Data Object Name" we see the name of each Data Object and in column E "Size (Bytes)" we see the size of each Data Object in bytes. In the column C "Parameters" we see the parameters of each Data Object and in the column "Format" we see The Format of the byte in bits.

For example, the Data Object named "FDR Config" which is the equivalent of dec ID "4" and its size is 1 byte, has the parameters "Volume DataBase period" and "Volume DataBase hour" and according to its byte "Format" described in column F, the first 3 bits represent the "FDR period" and its last 5 bits represent the "FDR hour".

Conversion of the Volume Index

As described in the Table of DOBJ, all the Volume Index and related to Volume data (Backflow Index / FDR ...) sent by the Intelis wSource are represented as Target Turn. From this unit, to display the data in the Liters / Gallon / Cubit feet unit, the decoder needs to apply the Unitary Value contained in Meter Config (DOBJ 151). The Itron Data Object (DOBJ) is a specific identifier used to represent various types of data sent by the Intelis wSource. Each DOBJ has a size and represents different data like volume index, water consumption, meter configuration, etc. These DOBJs are present in the payload frame in hexadecimal format and the decoder's role is parsing the payload to determine the list of DOBJs included on it. For example, in the payload frame "97043C3D9716003DC2060000", there are multiple DOBJs like Meter Config (DOBJ 97), Volume Index (DOBJ 3C), and Backflow Index (DOBJ 3D). Each of these DOBJs has a specific byte size and represents different data. Let's break down this example further:

Example: Payload Frame = 97043C3D9716003DC2060000

- DOBJ $97_{(16)} = 151_{(10)}$ => DOBJ "**Meter Config"** composed of 1 byte: $04_{(16)}$ =>Unitary Value = $0000100_{(2)} = 4_{(10)}$ => Unitary Value = 4 = 1L.
- DOBJ $3C_{(16)} = 60_{(10)}$ => DOBJ "**Volume Index**" composed of 4 bytes: $3D971600_{(16)}$ =>Volume Index = $0016973D_{(16)} = 1480509_{(10)}$ => Volume Index = 1480509 Target Turns.
- DOBJ $3D_{(16)} = 61_{(10)}$ => DOBJ "**Backflow Index**" composed of 4 bytes: $C2060000_{(16)}$ Volume Index = $000006C2_{(16)} = 1730_{(10)}$ => Backflow Index = 1730 Target Turns.

  The Intelis wSource has a Unitary Value of 1L, giving a Volume Index of 1480509L or 1480,509m^3 and a Backflow Volume of 1730L.

Water Consumption Calculation

As described in the Excel File of DOBJ, the Intelis wSource can send the Water Consumption of the Meter. This Water Consumption will be configured according to the End User Use Case. This configuration is defined by a periodicity (FDR Period) which can be 15min, Hourly, or Daily and an Hour (FDR Hour) for the case of Daily Period. Based on this configuration, the Intelis wSource will fill up a table of Water Consumption which will be transmitted via radio through delta interval. Based on the DOBJ used, this interval can be composed of 2 up to 24 steps of periodicity.

In the example of DOBJ 175: FDR 12 Delta S16, this DOBJ is composed of 12 intervals and each interval represents a step of 1 periodicity. Each interval will represent the delta of the Index between the previous step and the next one. In consequence, based on the Volume Global Index which represents the index from the last recording of Water Consumption, we can recompose the table of the last 12 intervals of periodicity.

N	N - N-1	N-1 - N-2	…	N-11 - N-12
Time of Record	Volume Delta between time of record and 1st previous step of periodicity	Volume Delta between 1st step of periodicity and 2nd step before		Volume Delta between 11th step of periodicity and 12th step before
Volume Index at N	Volume Index at N-1 = Volume Index at N - delta during this interval	Volume Index N-2 = Volume during previous interval - delta during this interval		Volume Index at N-12 = Volume during previous interval - delta during this interval

Fig.21 Screenshot from Itron's decoding datasheet showcasing the data intervals

The provided tables outline how to calculate water consumption over time by noting the 'delta' or change in volume index between successive periods. This is to say, for a given N (current reading time), you calculate the consumption for the interval by subtracting the volume at the previous step (N-1, N-2, ..., N-24, etc.) from the volume at the current step. This allows you to track consumption over an extended period by understanding the changes between one reading and the next, over the course of several intervals.

The decoding process would therefore involve:

1) Identifying the DOBJ for each set of data within your decrypted payloads. This will tell what type of data we are looking at (e.g., current index, backflow index, etc.).
2) Applying the Unitary Value to convert the target turns into a human-readable unit (e.g., liters, gallons).
3) Understanding the periodicity (FDR Period) to correctly interpret the time span each data point covers.
4) Reconstructing the water consumption table based on the deltas provided in the payload, using the current index as a starting point, and working backward through the intervals.

Overall, the process requires parsing the payload into these DOBJ components, understanding the format and size of each, and applying the correct mathematical operations to derive meaningful data.

So, for example, for the payload:

af040301000003000000000000000000000000000000000000000000000000000b004022000000007fff0000 we can see the first byte being "af" which is 175 in decimal, so we have a DOBJ 175 case. According to DOBJ 175 table, the first parameter refers to "Meter Config" which is described in the DOBJ 151 and has a size of one byte. So, in our case "04" which means that one target turn is equal to one Liter. Then we have the next four bytes, that represent the "Volume Database Global Index" which indicates the last record of the Global Volume Index N. It is encoded LSB first according to column F and it's measured in target turns. So, in our case 00000103(hex) which is 259 (dec)target turns. So, the last record of the Index N is 259 Liters. The next byte is about FDR Config (DOBJ 4) which we already discussed. The next 24 bytes indicate the last 12 Interval of data with step of 1 and it is LSB encoded as well. After that, the decoder still needs to search the payload

for other DOBJ values which in our case we have "b0" which is DOBJ 176, and repeat the process for DOBJ 176. After that, the payload is parsed from start to end, and the data are now stored in a new file in human-readable form.

The output string, we would see for that specific payload is:

['Liters', 253, 'hourly', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

We would understand that the total water consumption captured by this device is 253 and that no water consumption was captured in the last 12 hours.

Here's an overview of the steps:

- Open the decrypted_payload.txt file and read the Payloads from the file which are in hexadecimal format.
- For each payload, the content is parsed into separate DOBJ sections based on the information provided in the DOBJ Table.
- For each DOBJ section, identify the corresponding parameters and format.
- Convert each section according to its specified format, applying any necessary calculations (like unit conversions).
- Write the decoded, human-readable data into a new file, named decoded_data.txt.

```python
# Function to decode a single payload
def decode_payload(payload):
    if payload.startswith("af04"):
        decoded = ["Liters"]
        # Decoding the next 4 bytes from LSB to decimal
        liters = int.from_bytes(bytes.fromhex(payload[4:12]), byteorder='little')
        decoded.append(liters)

        if payload[12:14] == "03":
            decoded.append("hourly")
            # Processing the next 24 bytes
            hourly_data = payload[14:62]
            # Reversing the bytes for MSB encoding
            hourly_data_reversed = ''.join(reversed([hourly_data[i:i+2] for i in range(0, len(hourly_data), 2)]))
            # Splitting into 2-byte pairs and converting to decimal
            hourly_values = [int(hourly_data_reversed[i:i+4], 16) for i in range(0, len(hourly_data_reversed), 4)]
            decoded.extend(hourly_values)

        return decoded
    else:
        return None

def process_payloads(file_path, output_path):
    decoded_payloads = []
    with open(file_path, "r") as file:
        for line in file:
            match = re.search(r'af[0-9a-f]+', line)
            if match:
                payload = match.group(0)
                decoded = decode_payload(payload)
                if decoded:
                    decoded_payloads.append(decoded)

    # Writing the decoded values to a new file
    with open(output_path, "w") as file:
        for decoded in decoded_payloads:
            file.write(f"{decoded}\n")
```

Fig.22 The decoder in Python code

Conclusion

We are now able to receive water consumption data transmitted with LoRaWAN, decrypt the payload, and store the information in a Txt file locally. The next section is about storing the information in InfluxDB and visualizing it on Grafana.

3.5 DATA STORAGE AND VISUALIZATION

For the data to be stored and visualized properly instead of a txt file, we used the InfluxDB-Grafana combo as mentioned before. In this chapter, we will go through the setting up and configuring of both tools.

3.5.1 INFLUXDB

InfluxDB is an open-source time-series database that is optimized for high write and query loads. It is particularly suitable for use cases involving large amounts of timestamped data, such as IoT applications, monitoring, and real-time analytics. In this section, we will detail the steps taken to configure InfluxDB on an Ubuntu host.

Setting Up InfluxDB

The setup process involves creating an initial user, organization, and bucket. We executed the influx setup command to initialize these settings.

```
ginio@192 ~ % influx
NAME:
   influx - Influx Client

USAGE:
   influx [command]

HINT: If you are looking for the InfluxQL shell from 1.x, run "influx v1 shell"

COMMANDS:
   version              Print the influx CLI version
   write                Write points to InfluxDB
   bucket               Bucket management commands
   completion           Generates completion scripts
   query                Execute a Flux query
   config               Config management commands
   org, organization    Organization management commands
   delete               Delete points from InfluxDB
   user                 User management commands
   task                 Task management commands
   telegrafs            List Telegraf configuration(s). Subcommands manage Telegraf configurations.
   dashboards           List Dashboard(s).
   export               Export existing resources as a template
   secret               Secret management commands
   v1                   InfluxDB v1 management commands
   auth, authorization  Authorization management commands
   apply                Apply a template to manage resources
   stacks               List stack(s) and associated templates. Subcommands manage stacks.
   template             Summarize the provided template
   bucket-schema        Bucket schema management commands
   scripts              Scripts management commands
   ping                 Check the InfluxDB /health endpoint
   setup                Setup instance with initial user, org, bucket
   backup               Backup database
   restore              Restores a backup directory to InfluxDB
   remote               Remote connection management commands
   replication          Replication stream management commands
   server-config        Display server config
   help, h              Shows a list of commands or help for one command

GLOBAL OPTIONS:
   --help, -h  show help
ginio@192 ~ % export INFLUX_TOKEN=WqLYsURgNHHhZ88DaHoRfCwOruLdzsH4rsTwwnPUncwGYT1fTdWO_1ZDfpgIfLZhbl3EWdCsMQYB808jdtvuWg==
export INFLUX_ORG=TUC
ginio@192 ~ % influx bucket list

ID                  Name                Retention       Shard group duration    Organization ID         Schema Type
c96c63ce9a213df7    Water Consumption   infinite        168h0m0s                ff5e4cf9ffd651b7        implicit
d6350ecc9efd8262    _monitoring         168h0m0s        24h0m0s                 ff5e4cf9ffd651b7        implicit
bd9365a2af27b999    _tasks              72h0m0s         24h0m0s                 ff5e4cf9ffd651b7        implicit
077b083b71f8ace6    sensor_data         infinite        168h0m0s                ff5e4cf9ffd651b7        implicit
```

Fig.23 Instance from the terminal showcasing the InfluxDB setup

During the InfluxDB setup process, we entered several critical pieces of information to configure the database appropriately. First, we provided a username and password to create an admin user, ensuring secure access to the database. We then specified the organization name as "TUC" to identify the organization managing the database. To organize and store the incoming data, we created a bucket named "sensor_data." This bucket serves as the primary storage location for our

data entries. Finally, we set the retention period to "infinite," meaning that the data stored in the "sensor_data" bucket will be retained indefinitely, without any automatic deletion. This configuration ensures that our setup is secure, organized, and capable of preserving data for as long as necessary.

Configuring InfluxDB for Use with Flask

With InfluxDB installed and set up, we proceeded to configure it for use with our Flask application. This involved creating an authentication token and configuring the Flask application to use this token for data writes.

During the setup process, a token was automatically created. We noted down the token which has permission to read and write data to our sensor_data bucket and updated the Flask application to use the InfluxDB client with the generated token. Below is a snippet of the relevant Flask application configuration:

```
app = Flask(__name__)

# InfluxDB details
token = "ouRZEzPTSYJk7yNuWT9tvD0E8Aq0RQACiBKLzhM4L44ccC0H5YehMByWBZ00SxP1JwXL9QLKaAKEHHXiqZADBA=="
org = "TUC"
bucket = "sensor_data"

# Initialize InfluxDB Client
client = InfluxDBClient(url="http://localhost:8086", token=token, org=org)
write_api = client.write_api(write_options=SYNCHRONOUS)
```

Fig.24 Enabling the Flask application to write data on InfluxDB

In this configuration (Fig.24):

- We used the InfluxDBClient class to create a client instance.
- The URL parameter specifies the address of the InfluxDB instance.
- The token parameter is used for authentication.
- The org parameter specifies the organization name.
- The bucket parameter specifies the bucket where data will be stored.

We then updated the Flask application to write data points to InfluxDB. Each data point included various fields such as the total consumption and hourly measurements. Here is an example of how data points were written:

```
# Prepare data for InfluxDB
point = Point("sensor_reading")\
    .tag("unit", measurement_unit)\
    .tag("interval", interval)\
    .tag("sensor", dev_eui)\
    .field("total_consumption", total_consumption)\
    .time(datetime.utcnow(), WritePrecision.NS)

for i, val in enumerate(hourly_consumption, start=1):
    point.field(f"hour_{i}", val)

# Write points to InfluxDB
write_api.write(bucket=bucket, org=org, record=point)
```

Fig.25 Configuring how the data will be written on InfluxDB

In this snippet (Fig.25):

- Point is used to create a new data point.
- .tag() is used to add metadata to the point.
- .field() is used to add the actual data values.
- write_api.write() is used to write the point to the specified bucket in InfluxDB.

Outcome

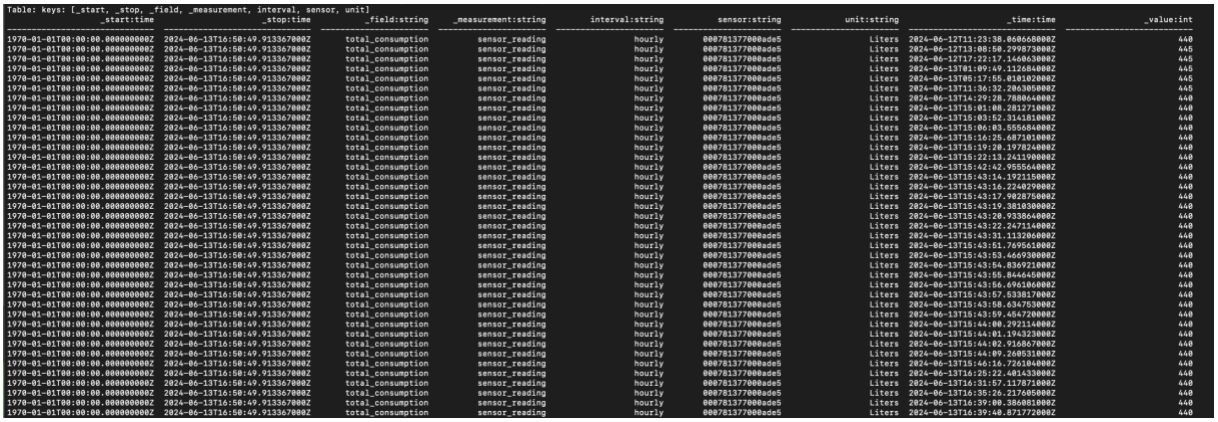


Table: keys: [_start, _stop, _field, _measurement, interval, sensor, unit]

_start:time	_stop:time	_field:string	_measurement:string	interval:string	sensor:string	unit:string	_time:time	_value:int
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-12T11:23:38.060668000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-12T13:08:50.299873000Z	445
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-12T17:22:17.164063000Z	445
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T01:09:49.112684000Z	445
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T05:17:55.010102000Z	445
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T11:36:32.206305000Z	445
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T14:29:28.788064000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:01:08.281271000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:03:52.314181000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:06:03.555684000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:16:25.687101000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:19:20.197024000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:22:13.241190000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:42:42.955564000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:14.192115000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:16.224029000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:17.902875000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:19.381030000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:20.933864000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:22.247114000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:31.113206000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:51.769561000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:53.466930000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:54.836921000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:55.844445000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:56.696104000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:67.533817000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:58.634753000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:43:59.456728000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:44:00.292114000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:44:01.194323000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:44:02.910867000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:44:09.260531000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T15:46:16.726104000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T16:25:22.401433000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T16:31:57.117071000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T16:35:26.217605000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T16:39:00.386081000Z	440
1970-01-01T00:00:00.000000000Z	2024-06-13T16:50:49.913367000Z	total_consumption	sensor_reading	hourly	00078137700ade5	Liters	2024-06-13T16:39:40.871772000Z	440

Fig.26 Instance of InfluxDB running

The image above shows a snapshot of the InfluxDB application running. The table contains a collection of time-series data entries, organized with several key attributes:

- _start: The beginning of the time interval for the data points.
- _stop: The end of the time interval for the data points.
- _time: The exact timestamp for each data point within the interval.
- field_string: The specific field being measured, in this case, "total_consumption."
- measurement: The measurement type, is labeled as "sensor_reading."
- interval_string: The interval at which data is recorded, set to "hourly."
- sensor_string: The unique identifier for the sensor collecting the data.
- unit_string: The unit of measurement for the recorded data, which is "Liters" in this instance.
- time: Another column for the timestamp of the data point.
- value_int: The recorded value of the measurement at the given timestamp.

With that, we managed to set up InfluxDB and store data there, which is a crucial step toward the end goal- visualization. The next and final step of our implementation is exactly about that.

3.5.2 GRAFANA

In this section, we will detail the steps taken to set up and configure Grafana on an Ubuntu server to visualize data collected from a LoRaWAN network server managed by ChirpStack. The Grafana setup allows for real-time monitoring and visualization of sensor data stored in InfluxDB.

Once Grafana was installed and running, the next step was to configure it for visualizing the data stored in InfluxDB. The configuration steps included:

1) Access Grafana

Logged in the address http://localhost:3000/. The default login credentials were admin for both the username and password. Upon first login, we were prompted to change the default password.

2) Add InfluxDB as a Data Source
   - Navigate to the Grafana homepage.
   - Go to Configuration (the gear icon) on the left-hand side menu, and click on Data Sources.
   - Click the Add data source button.
   - Select InfluxDB from the list of available data sources.

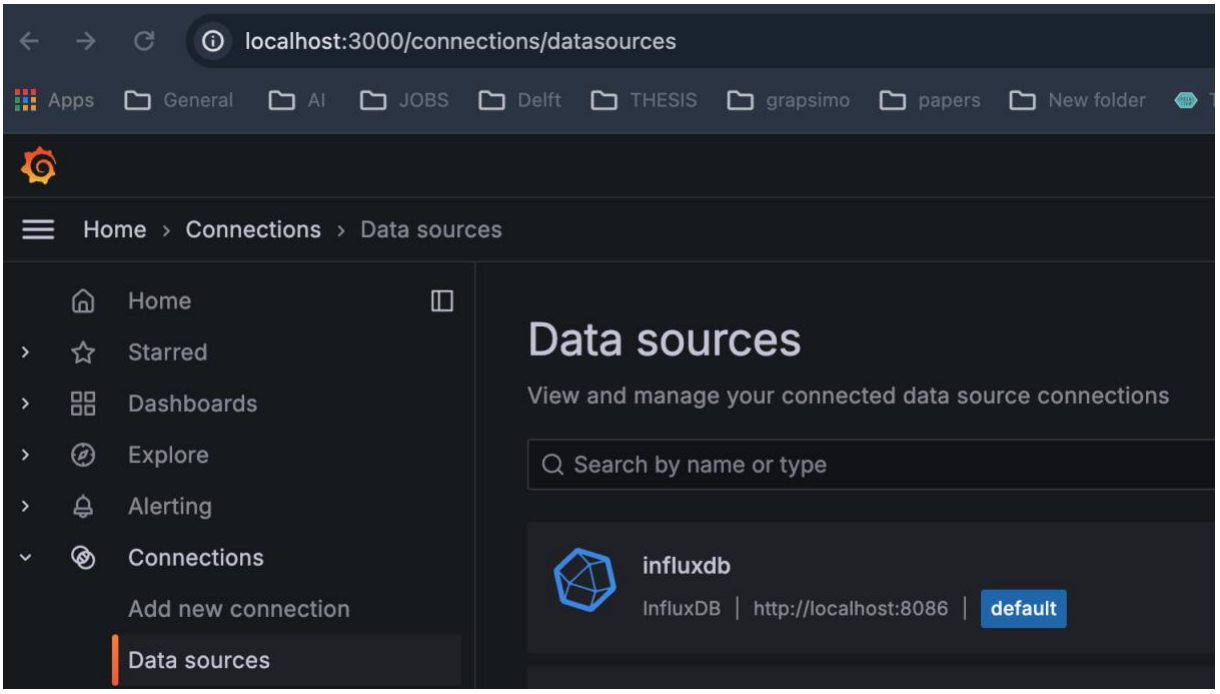


Fig. 27 Setting up Grafana

3) Fill in the details for the InfluxDB configuration.
   - Name: Entered the data source a name (e.g., InfluxDB).
   - URL: Entered http://localhost:8086 (assuming InfluxDB is running on the same server as Grafana).
   - Database: Entered the name of our database (sensor_data).
   - Token: Entered the authentication token obtained during the InfluxDB setup.
   - Organization: Entered our InfluxDB organization name. (TUC).

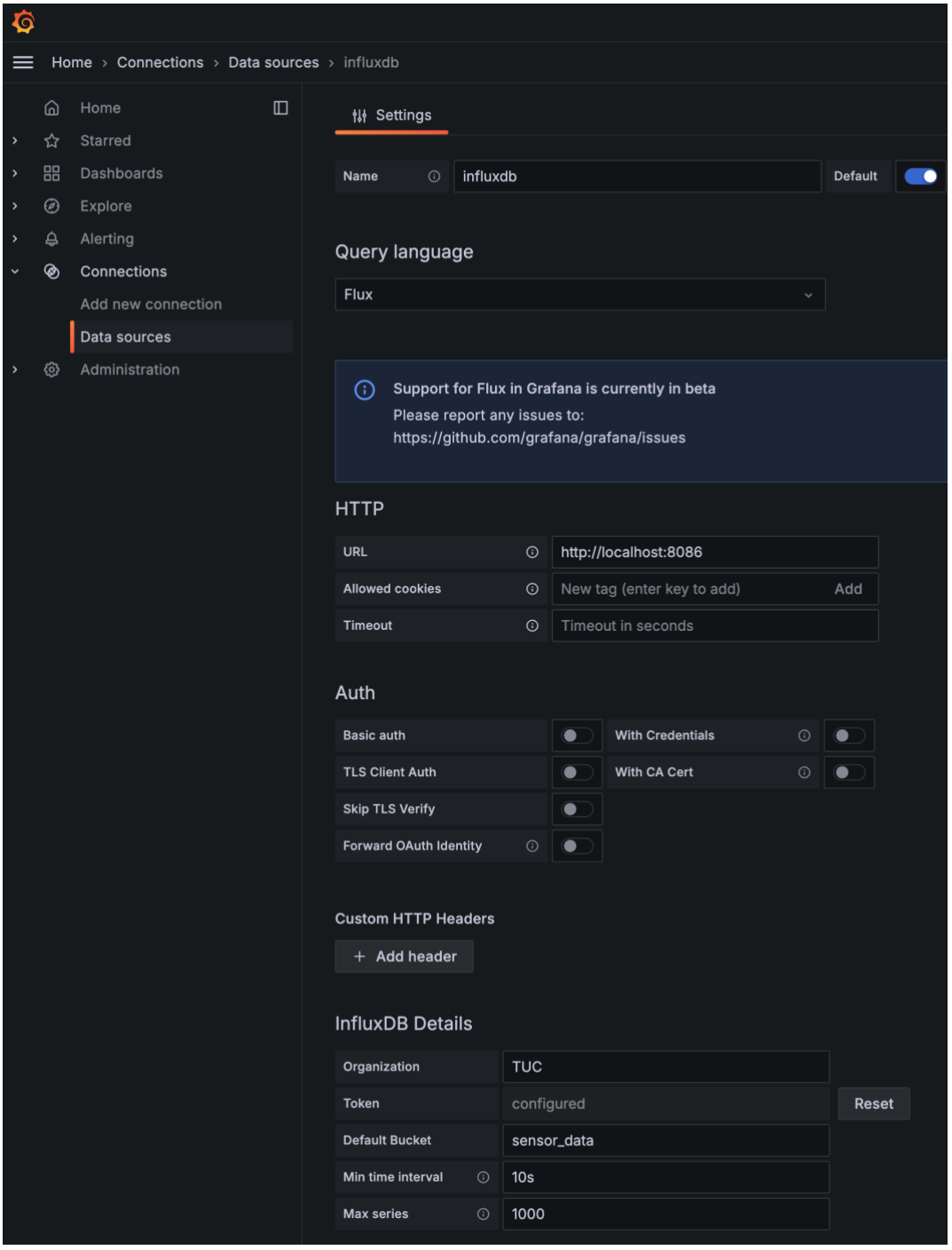


Fig.28 Integrating with Influx

4) Creating Dashboards

With the data source successfully configured, the next step was to create dashboards to visualize the sensor data. To create dashboards in Grafana to visualize stored data from InfluxDB, begin by clicking on the “+” icon located on the left-hand side menu and selecting the option for Dashboard. Next, choose the configured InfluxDB data source from the list of available sources. After selecting the data source, proceed to write a query that fetches the required data from InfluxDB. This query should be tailored to retrieve the specific metrics or information you wish to visualize on your dashboard. For example :

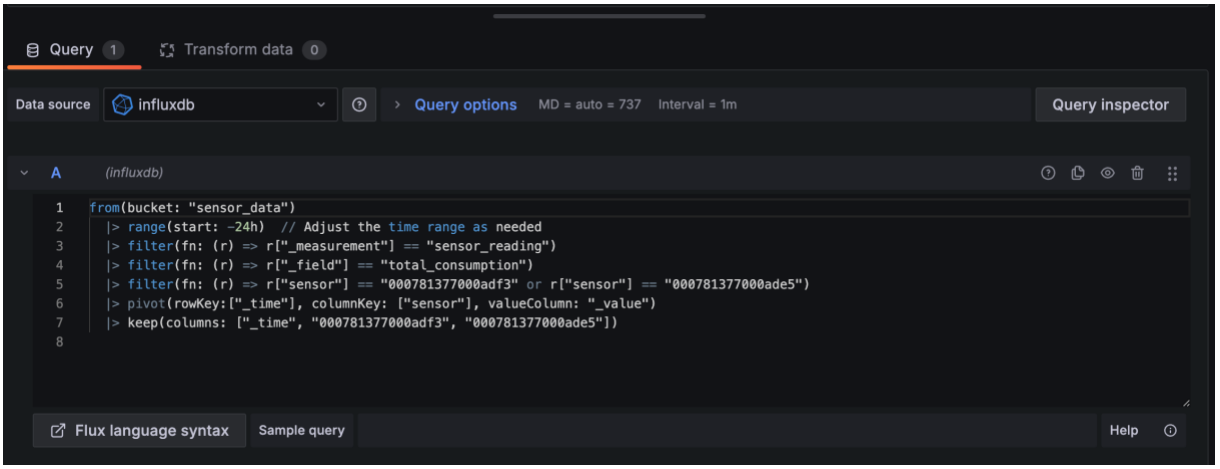


Fig.29 Flux query written in Grafana environment

This query retrieves data from the "sensor_data" bucket for the last 24 hours, filters it to include only relevant sensor readings for total consumption, pivots the data so that each sensor's values are in separate columns, and keeps only the time and sensor value columns for the specified sensors.

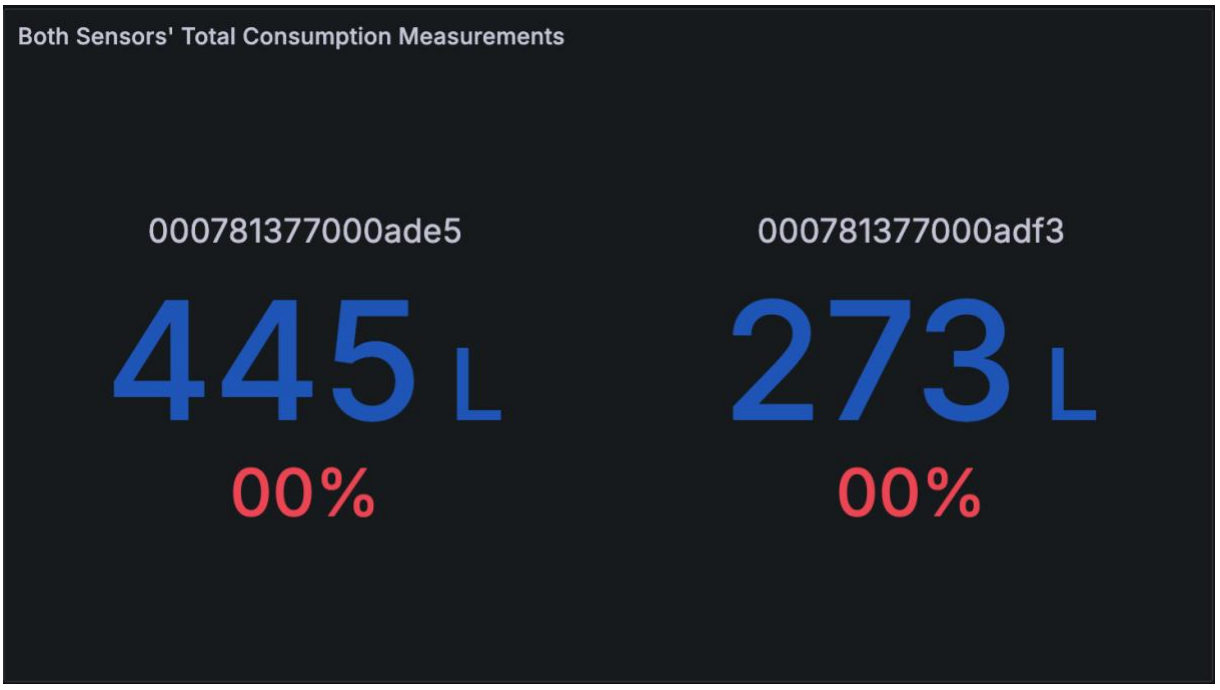


Fig.30 Output of the query displayed above

As already mentioned, one of the key features of InfluxDB is the time-series data storing and thus we can leverage that on Grafana by creating another dashboard and writing another Flux Query for that task. For example :

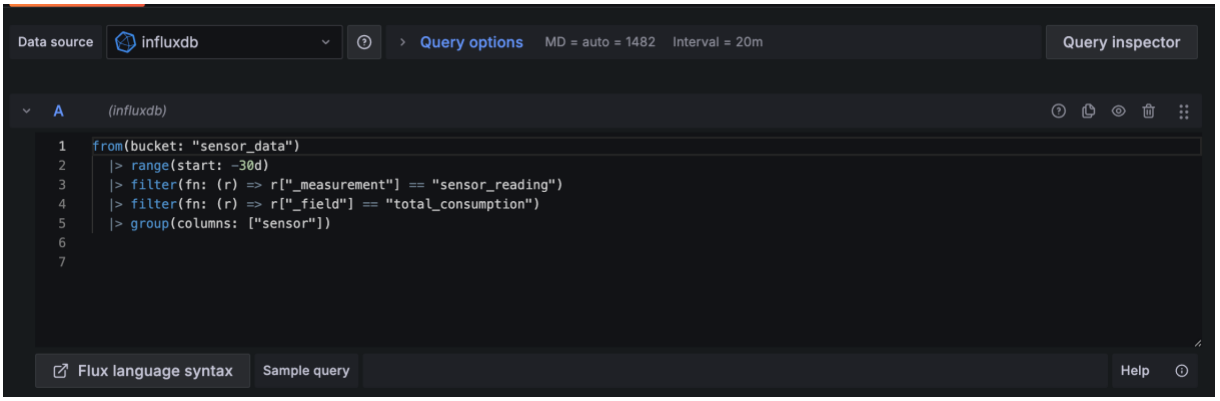


Fig.31 Flux Query for time series display

This query retrieves data from the "sensor_data" bucket for the past 30 days, filters it to include only "sensor_reading" measurements and "total_consumption" fields, and groups the data by the "sensor" column for further analysis or visualization.

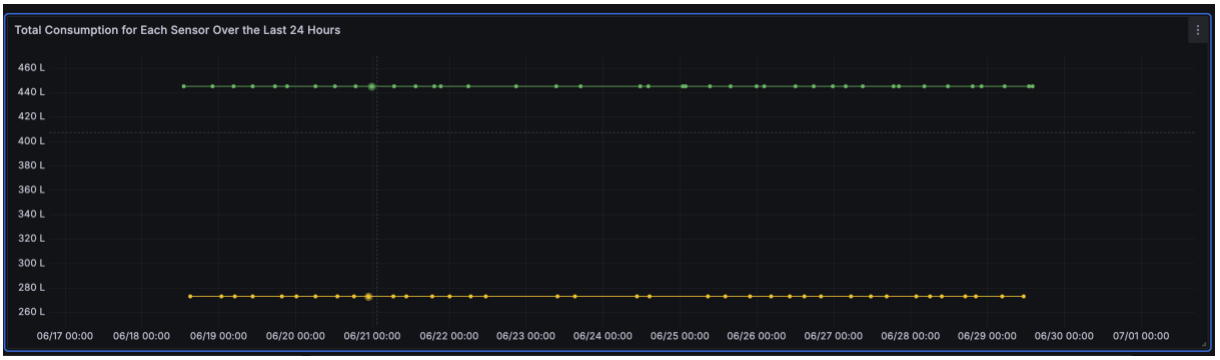


Fig.32 Output of the query displayed above

Here's a detailed explanation of each part of the last query:

- from(bucket: "sensor_data"): This line specifies the data source, indicating that the query should retrieve data from the "sensor_data" bucket. In InfluxDB, a bucket is a named location where time-series data is stored.
- |> range(start: -30d): This function narrows down the time range of the data to be queried. The start: -30d argument means that the query will only include data from the last 30 days up to the current time.
- |> filter(fn: (r) => r["_measurement"] == "sensor_reading"): This filter function selects only the records where the _measurement field equals "sensor_reading". Measurements are logical groupings of records in InfluxDB, similar to tables in traditional databases.
- |> filter(fn: (r) => r["_field"] == "total_consumption"): This additional filter function further refines the data by selecting records where the _field is "total_consumption". Fields in InfluxDB are key-value pairs that hold the actual data values.
- |> group(columns: ["sensor"]): Finally, this function groups the resulting data by the "sensor" column. Grouping by "sensor" means that the data will be organized into groups

based on the distinct values of the "sensor" column, allowing for more granular analysis and visualization of data from each sensor.

Interpretation of Dashboards

The dashboard of Fig.30 informs us that the total consumption of each water meter is 445L and 273L respectively while the 0% indicates the percentage of the volume change from the last measurement.

The dashboard of Fig.32 informs us that the total consumption of each water meter remained unchanged from 19/6 since its last recording at 29/6. This makes sense since we did not have any water-meter under flowing water those days, and thus our system works as supposed.

Lastly, it is worth mentioning that for every new payload received, the dashboards refresh automatically, and it is possible to have as many dashboards as needed functioning at the same time.

Fig.33 Monitoring 4 dashboards at the same time.

4. RESULTS AND DISCUSSION

4.0 INTRODUCTION

The purpose of this chapter is to present and discuss the results of the implemented system for managing and visualizing sensor data transmitted with LoRaWAN. The chapter will cover the system's performance in terms of data collection and end with a discussion of future work.

4.1 NETWORK PERFORMANCE ANALYSIS

Network performance analysis is a crucial aspect of any IoT deployment, including the one involving the Intelis wSource water meter and LoRaWAN technology. This section provides an in-depth analysis of the network performance, evaluating various metrics such as RSSI, SNR, packet loss rate, and data transmission intervals. The analysis aims to understand the reliability, efficiency, and robustness of the LoRaWAN network under different conditions. All the metrics discussed in this chapter, take into consideration: spreading factor (SF) = 7 and Adaptive Data Rate (ADR) = 5.

Key Performance Metrics

1) Received Signal Strength Indicator (RSSI)

RSSI measures the power level of the received signal. In the context of LoRaWAN, it indicates how well the gateway is receiving the signal from the end device.

2) Signal-to-Noise Ratio (SNR)

SNR is the ratio of the desired signal to the background noise level. A higher SNR indicates a cleaner and stronger signal.

3) Packet Loss Rate

Packet loss occurs when one or more packets of data fail to reach their destination. It can be measured by comparing the number of packets sent by the device to the number received by the gateway. Monitoring the packet loss rate helps in identifying communication reliability issues.

4) Data Transmission Intervals

This metric evaluates the frequency of data transmissions from the devices. Consistent intervals indicate stable network performance, whereas irregular intervals might suggest network congestion or other issues.

The performance metrics are influenced by several environmental and operational factors:

- Distance and Obstructions: The distance between the device and the gateway, and the presence of physical obstructions (buildings, trees, etc.), can significantly affect signal strength and quality.
- Environmental Conditions: Weather conditions such as rain, humidity, and temperature can impact signal propagation.
- Data Rates and Spreading Factors: Higher spreading factors generally result in a longer range but lower data rates.

4.2 PACKET LOSS RATE ANALYSIS

Since the Intelis wSource sends the water consumption registered for the last 12 hours and every uplink has a 6-hour phase difference from the previous one, that would mean that we need to receive at least 3 uplinks daily or 2 non-consecutive to not miss any data point. The following images are taken from the Chirpstack which provides the necessary graphs to monitor your system's performance. As we said, we have deployed two Intelis wSource devices in our system with DevEUI's 000781377000adf3 and 000781377000ade5 respectively. The graphs illustrate the received payloads per day. Both devices were initialized on June 5.

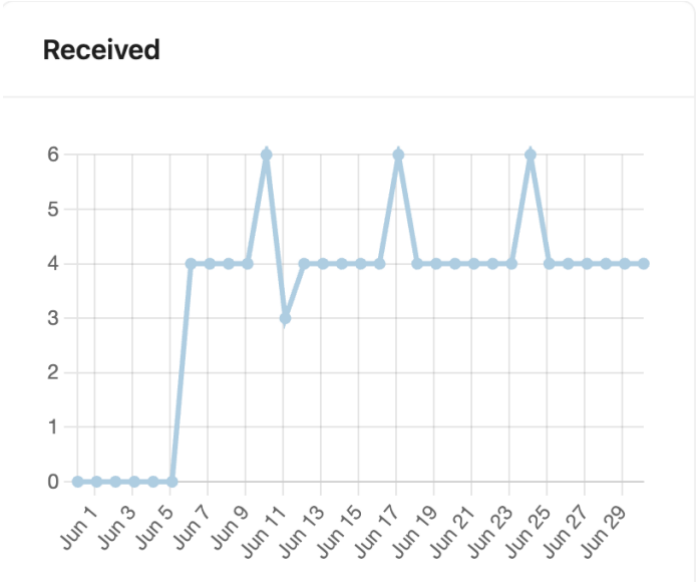


Fig.34 Number of daily payloads received from 000781377000adf3

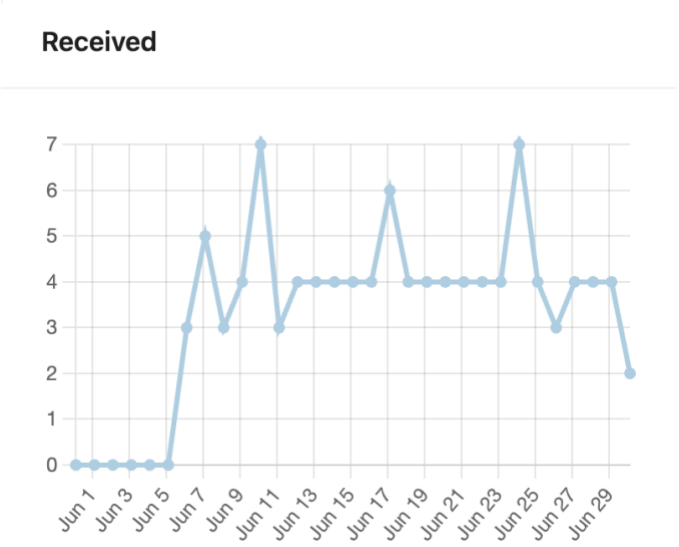


Fig.35 Number of daily payloads received from 000781377000ade5

4.3 RSSI ANALYSIS

The Received Signal Strength Indicator (RSSI) is a critical parameter in wireless communication that measures the power level of the received signal at the receiver's end. It plays a vital role in determining the quality of the wireless link and overall network performance. This section delves into the significance of RSSI, its measurement, and its impact on network performance.

RSSI is a measurement of the power present in a received radio signal. It is typically expressed in decibels relative to a milliwatt (dBm). The RSSI value provides insight into the signal strength that a device is experiencing from a transmitter, such as a Wi-Fi router, cellular tower, or a LoRa gateway.

RSSI is measured by the hardware of the receiver, such as a network interface card (NIC) in a computer, a mobile phone's radio, or an IoT device's transceiver. The RSSI value is usually a negative number; the closer the value is to zero, the stronger the signal. For example, an RSSI of -30 dBm indicates a very strong signal, while an RSSI of -100 dBm indicates a weak signal.

In technologies like LoRaWAN, RSSI is used in conjunction with other parameters like Signal-to-Noise Ratio (SNR) to adapt data rates dynamically. This ensures efficient use of network resources and maintains reliable communication even in challenging conditions.

Factors Affecting RSSI

Distance: RSSI decreases with increasing distance between the transmitter and receiver. This is due to the attenuation of the signal as it travels through the medium.

Obstructions: Physical obstructions like walls, buildings, and trees can attenuate the signal, leading to lower RSSI values. The type and density of the material also play a significant role.

Interference: Interference from other electronic devices operating on the same or nearby frequencies can affect RSSI. This is common in environments with multiple Wi-Fi networks or industrial areas with various radio-emitting devices.

Antenna Characteristics: The design and orientation of the antennas used by the transmitter and receiver influence RSSI. Higher gain antennas can improve RSSI by focusing the signal in a specific direction.

Interpreting RSSI Values

Strong Signal: RSSI values between -30 dBm and -67 dBm indicate a strong and reliable signal. This range is ideal for high-speed data transmission and real-time applications.

Moderate Signal: RSSI values between -68 dBm and -80 dBm indicate a moderate signal. Communication is typically reliable, but there might be occasional issues with higher data rates.

Weak Signal: RSSI values between -81 dBm and -90 dBm indicate a weak signal. There might be significant communication issues, including packet loss and lower data rates.

Very Weak Signal: RSSI values below -90 dBm indicate a very weak signal. Communication is likely to be unreliable or fail altogether.

Results

Having discussed the RSSI and how to interpret it, let's have a look at the graphs illustrating the actual RSSI values of our system.

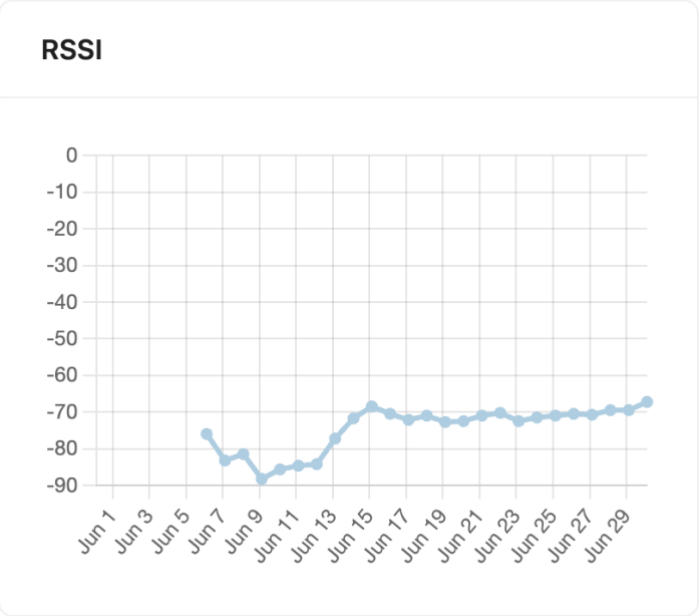


Fig.36 RSSI values for 000781377000adf3 over time.

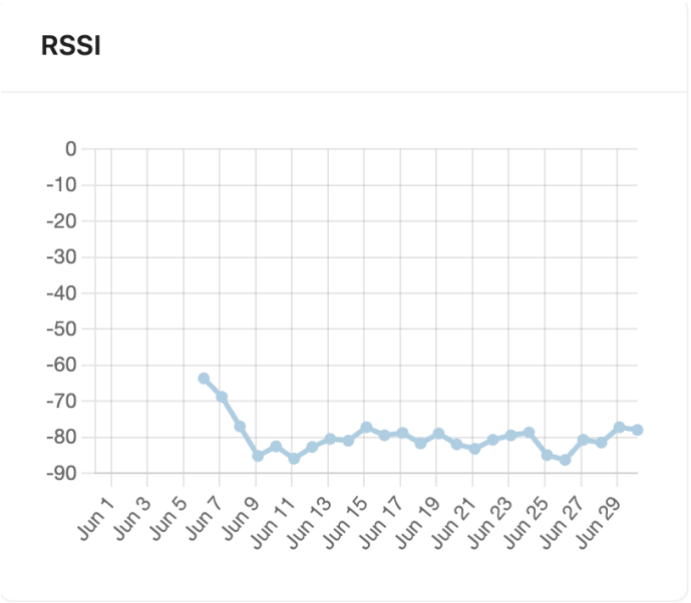


Fig.37 RSSI values for 000781377000ade5 over time.

The RSSI analysis for both devices reveals that they experience similar initial challenges with signal strength, likely due to environmental factors or initial setup conditions. Over time, both devices achieve stable RSSI values around -70 dBm to -80 dBm, indicating good signal strength and consistent performance.This consistency in RSSI values is crucial for reliable data transmission in LoRaWAN networks. The minor differences between the devices highlight the importance of precise positioning and environmental considerations in network performance.

4.4 SNR ANALYSIS

The Signal-to-Noise Ratio (SNR) is a critical parameter in wireless communication systems that measures the quality of a received signal. SNR is defined as the ratio of the power of the desired signal to the power of the background noise. It is usually expressed in decibels (dB).

importance of SNR in Wireless Communication

Quality of Communication: A higher SNR indicates a cleaner and clearer signal, which translates to better quality of communication. High SNR values mean that the signal can be distinguished from the noise, leading to fewer errors in data transmission.

Data Rate: The data rate of a communication system can be directly affected by SNR. Higher SNR allows for higher data rates because the system can use higher-order modulation schemes, which transmit more bits per symbol.

Reliability: Systems with higher SNR are more reliable as they are less susceptible to interference and noise. This is crucial in environments with multiple sources of electromagnetic interference.

Range: SNR also impacts the effective range of a wireless communication system. Higher SNR values allow for greater transmission distances without loss of signal quality.

Factors Affecting SNR

Distance: The distance between the transmitter and receiver significantly affects SNR. As the distance increases, the signal strength decreases, leading to a lower SNR.

Interference: Electromagnetic interference from other devices can increase the noise level, thereby reducing the SNR.

Environmental Conditions: Physical obstructions such as buildings, trees, and weather conditions can attenuate the signal, affecting the SNR.

Device Quality: The sensitivity and quality of the receiving device's antenna and internal components can influence the SNR.

SNR in LoRaWAN Networks

In LoRaWAN networks, SNR plays a vital role in ensuring efficient and reliable data transmission between end devices and gateways. The Adaptive Data Rate (ADR) mechanism in LoRaWAN relies on SNR, among other factors, to optimize the data rate, transmission power, and frequency settings of the end devices. This ensures that devices can communicate effectively while conserving battery life.

Positive SNR: Indicates that the signal is stronger than the noise. LoRaWAN devices with positive SNR can operate at higher data rates.

Negative SNR: Indicates that the noise level is higher than the signal strength. This is common in challenging environments, and the system may need to lower the data rate to maintain reliable communication.

Results

Having discussed the RSSI and how to interpret it, let's have a look at the graphs illustrating the actual RSSI values of our system.

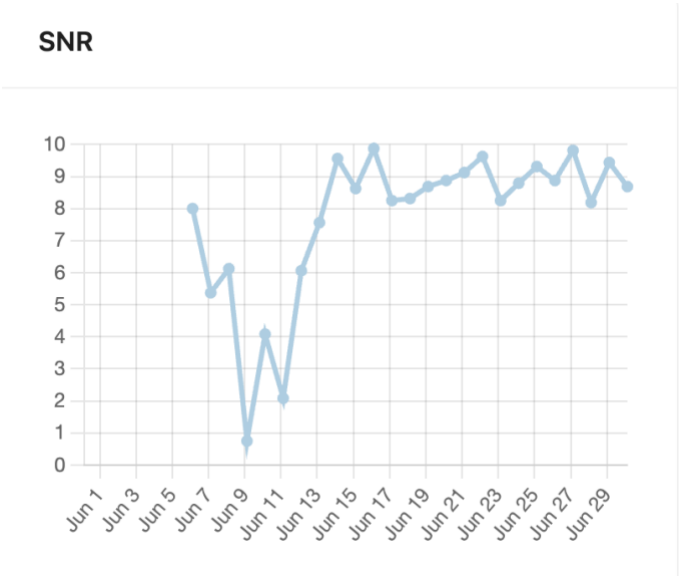


Fig.38 SNR values for 000781377000adf3 over time.

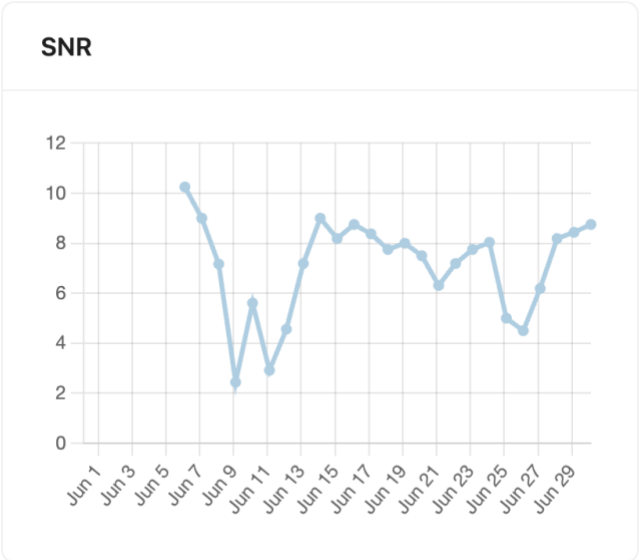
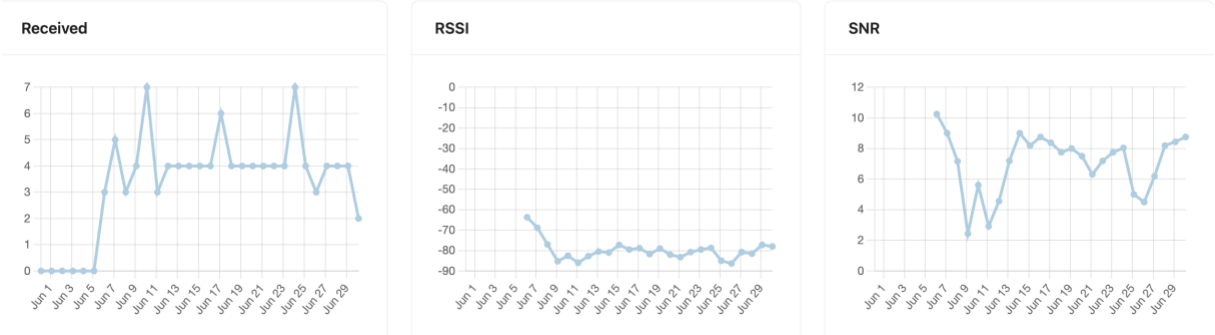


Fig.39 SNR values for 000781377000ade5 over time.

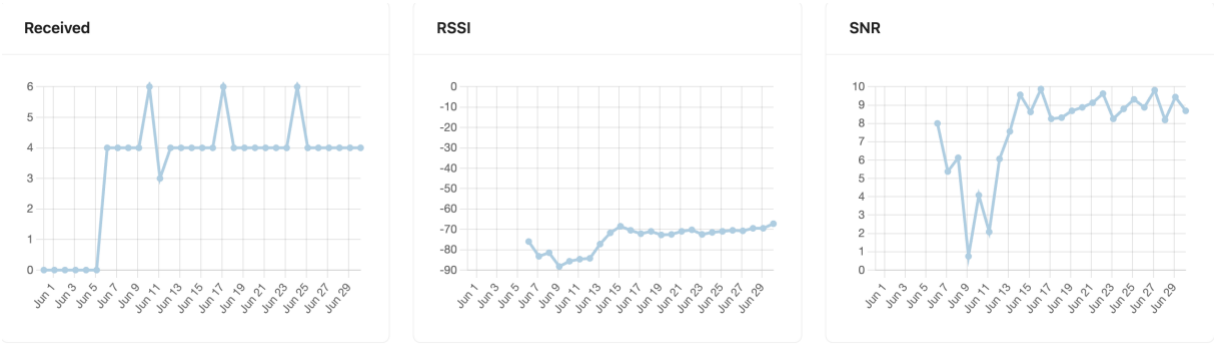
The analysis of the SNR graphs suggests that while both devices are subject to environmental conditions causing fluctuations in signal quality, Device 000781377000adf3 demonstrates more stability in its SNR values. Device 000781377000ade5, although having higher peak SNR values at times, also experiences more significant drops, indicating less consistent signal quality. For both devices, maintaining SNR values above 5 dB for the majority of the time suggests that they generally operate within acceptable limits for reliable communication. However, the period around June 7th to June 11th shows a clear degradation in signal quality, which indicates poorer environmental conditions during that period.

4.5 COMPARATIVE ANALYSIS

hydrometer-2 (000781377000ade5)



Hydrometer-1 (000781377000adf3)



Similarities

Both devices exhibit similar trends in received packets, with initial increases and subsequent fluctuations around a stable range of 4-6 packets per day. Both devices show improvement in RSSI values around mid-June. SNR values for both devices also show significant fluctuation in the early part of June but stabilize towards the end of the month.

Differences

Device 1 has slightly better RSSI and SNR values overall, indicating a potentially better reception or less interference. However, the peaks in received packets are more pronounced in Device 2, with slightly higher maximum values compared to Device 1.

Conclusion

Both devices show stable performance with similar trends in received packets, RSSI, and SNR. The slight differences in RSSI and SNR values suggest that Device 1 might have a marginally better network performance, potentially due to minor differences in positioning or hardware characteristics. However, both devices indicate good overall performance with stable and improving signal quality over the month of June.

5. CONCLUSION

This thesis aimed to explore the development and deployment of a LoRaWAN-based hydro-meter system, demonstrating the potential of IoT technologies in water management and monitoring. The primary objectives were to design a reliable, efficient, and scalable system for real-time water usage monitoring and to assess its performance in various environments.

The research findings indicate that the LoRaWAN-based hydro-meter system successfully meets the set objectives, providing accurate and timely data on water consumption. The system's performance was validated through extensive testing, showing reliable data transmission over long distances with minimal power consumption. The study also highlighted the system's capability to operate effectively in both urban and rural settings, making it a versatile solution for diverse applications.

Despite the successes, the study faced certain limitations, such as the potential for interference in dense urban areas and the need for robust security measures to protect data integrity. Addressing these limitations through future work, including sustainability studies, urban vs. rural deployment comparisons, long-term testing, and distance testing, will further enhance the system's capabilities and reliability.

The implications of this research are significant for the field of smart water management. Practically, the system can help utilities and consumers monitor and manage water usage more efficiently, contributing to water conservation efforts. Theoretically, the research adds to the growing body of knowledge on IoT applications in environmental monitoring, providing a foundation for future innovations.

In conclusion, the development of the LoRaWAN-based hydro-meter system marks a significant advancement in smart water management technologies. The insights gained from this study pave the way for further research and development, promising more efficient and sustainable water management solutions in the future. Continued exploration and refinement of this technology will undoubtedly contribute to the overarching goal of building smarter, more sustainable cities.

6. FUTURE WORK

The development and deployment of the LoRaWAN-based hydro-meter system have demonstrated the potential of IoT technologies in water management and monitoring. However, several areas for future work can enhance the capabilities, efficiency, and reliability of the system:

Sustainability Studies. Conducting studies to assess the environmental benefits of the system is crucial. These studies should focus on:

- Water Conservation: Analyzing how the system contributes to reducing water wastage and promoting sustainable water usage.
- Energy Consumption: Evaluating the energy efficiency of the system, especially the power consumption of devices and gateways. This includes exploring low-power communication protocols and energy harvesting techniques.

Urban vs. Rural Deployment. Comparing the performance of the system in different environments can provide valuable insights:

- Urban Settings: Identifying the specific challenges such as interference from buildings and high-density networks. Investigating how the system integrates with other urban infrastructure.
- Rural Settings: Understanding the range and coverage issues in sparse areas, and evaluating the system's reliability in remote locations.

Long-Term Testing. Long-term monitoring is essential to ensure the robustness of the system:

- Durability: Assessing the durability of devices and gateways over extended periods under various environmental conditions.
- Performance: Monitoring the system's performance metrics such as packet loss, latency, and battery life to identify and address any long-term issues.

Distance Testing. Analyzing the system’s performance over various distances from the gateway can help optimize range and coverage:

- Range Efficiency: Testing the efficiency of data transmission at different distances to determine the optimal range for reliable communication.
- Signal Quality: Evaluating how factors like RSSI and SNR vary with distance, and how this affects the overall performance and reliability of the system.

The implementation of the LoRaWAN-based hydro-meter system is a significant step towards smart water management. Future work in the areas mentioned above will help in overcoming current limitations, enhancing the system's capabilities, and ensuring its sustainability and scalability. Continued research and development will pave the way for more efficient and effective water management solutions, contributing to the overall goal of smart and sustainable cities.

BIBLIOGRAPHY

[1] X. V. P. T.-P. B. M. J. M.-S. T. W. Ferran Adelantado, "Understanding the Limits of LoRaWAN," *IEEE Communications,* 2017.

[2] X. Yang, E. Karampatzakis, C. Doerr and F. Kuipers, "Security Vulnerabilities in LoRaWAN," in *IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2018.

[3] P. Š. A. M. A. J. d. C. S. Joel Rodrigues, "LoRaWAN - A Low Power WAN Protocol for Internet of Things: a Review and Opportunities," 2017.

[4] D. B. A. L. Evgeny Khorov, "On the Limits of LoRaWAN Channel Access," Moscow, 2016.

[5] A. I. P. a. A. Lavric, "LoRaWAN communication protocol: The new era of IoT," in *14th International Conference on DEVELOPMENT AND APPLICATION SYSTEMS*, Suceava, 2018.

[6] 4. A. N. B. P. S. H. E. M.-C. a. G. R. H. Mroue1, "Analytical and Simulation study for LoRa Modulation," 2018.

[7] M. C. A. Elzanaty, "On the LoRa Modulation for IoT: Waveform Properties and Spectral Analysis," *EEE INTERNET OF THINGS JOURNAL,* 2019.

[8] M. Pettissalo, "Performance of a low-power wide-area network based on LoRa technology: Doppler robustness, scalability, and coverage," *International Journal of Distributed Sensor Networks,* 2017.

[9] U. Raza, "Low Power Wide Area Networks: An Overview," 2017.

[10] T. Voigt and A. Juan M., "Do LoRa Low-Power Wide-Area Networks Scale?," in *MSWiM*, Malta, 2016.

[11] Georgiou, Orestis; Raza, Usman, "Low Power Wide Area Network Analysis: Can LoRa Scale?," 2017.

[12] B. Norbert and K. Fernando , "LoRaWAN in the Wild: Measurements from The Things Network," 2017.

[13] J. Petajajarvi, A. Roivainen and K. Mikhaylov, "On the Coverage of LPWANs: Range Evaluation and Channel Attenuation Model for LoRa Technology," in *14th International Conference on ITS Telecommunications*, 2016.

[14] M. Cattani, C. B. Alberto and K. Romer, "An Experimental Evaluation of the Reliability of LoRa Long-Range Low-Power Wireless Communication," *Journal of Sensor and Actuator Networks,* 2017.

[15] R. Brecht, M. Wannes and P. Sofie, “Range and Coexistence Analysis of Long Range Unlicensed Communication,” 2016.

[16] J. Haxhibeqiri, V. d. A. Floris, I. Moerman and H. Jeroen, “LoRa Scalability: A Simulation Model Based on Interference Measurements,” *Sensors,* 2017.

[17] J. Haxhibeqiri, Eli De Pooter, Ingrid Moerman and Jeroen Hoebeke, “A Survey of LoRaWAN for IoT: From Technology to Application,” *Sensors,* p. 38, 2018.

[18] Mehmet Ali Erturk, Muhammed Ali Aydin, Muhammet Talha Buyukakkaslar and Hayrettin Evirgen, “ A Survey on LoRaWAN Architecture, Protocol and Technologies,” *Future Internet,* p. 34, 2019.

[19] Lluis Casals, Bernat Mir, Rafael Vidal and Carles Gomez, “Modeling the Energy Performance of LoRaWAN,” *Sensors,* 2017.

[20] M. Almuhaya, Waheb Jabbar , Noorazliza Sulaiman and Suliman Abdulmalek, “A Survey on LoRaWAN Technology: Recent Trends, Opportunities, Simulation Tools and Future Directions,” *electronics,* 2022.

[21] Emekcan Aras, Gowri Sankar Ramachandran, Piers Lawrence and Danny Hughes, “Exploring The Security Vulnerabilities of LoRa”.

[22] Michele Luvisotto , Federico Tramarin , Lorenzo Vangelista and Stefano Vitturi, “On the Use of LoRaWAN for Indoor Industrial IoT Applications,” 2018.

[23] L. Vangelista, “Frequency Shift Chirp Modulation: the LoRaTM Modulation,” 2017.