Enabling Edge Computing over LoRaWAN: A Device-Gateway Coordination Protocol

Ivan Fardin
fardin.1747864@studenti.uniroma1.it
DIAG, Sapienza University of Rome
Rome, Italy

Stefano Milani
milani@diag.uniroma1.it
DIAG, Sapienza University of Rome
Rome, Italy

Francesca Cuomo
francesca.cuomo@uniroma1.it
DIET, Sapienza University of Rome
Rome, Italy
CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni
Parma, Italy

Ioannis Chatzigiannakis
ichatz@diag.uniroma1.it
DIAG, Sapienza University of Rome
Rome, Italy
CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni
Parma, Italy

ABSTRACT

The freedom of the LoRaWAN license-free ad-hoc deployment model can significantly reduce the complexity of network deployment, however, it introduces certain problems that hinder network scalability and performance. First, Network Gateways simply forward frames to the centralised Network Server without the possibility of becoming Edge computing processing elements. Second, Network Gateways may have overlapping areas of network coverage (a) resulting in an increase of network traffic at the network backbone as frames are relayed to the Network Server multiple times, (b) may cause unexpected frame collisions and duty-cycle exhaustion. In this paper, a novel decentralised algorithm is presented that assigns each IoT Device to a single Network Gateway so that (a) duplicate message deliveries are completely avoided, (b) Network Gateways can become an intermediate operations layer between the IoT devices and the Network Server providing computational and storage resources. The proposed protocol is implemented in the OMNeT++ simulator and evaluated based on datasets collected from long-term real-world deployments. To improve the accuracy of the experiments in large-scale and dense urban deployments the OMNeT++ LoRa interference model is extended by considering the non-perfect orthogonality of the LoRa Spreading Factors and the channel adjacency. The performance evaluation highlights the benefits of the distributed approach proposed here and provides valuable indications on the overall performance of the network.

CCS CONCEPTS

• **Computer systems organization → Sensor networks**; **Sensors and actuators**; • **Networks → Network protocol design**; **Network experimentation**; **Network simulations**.

KEYWORDS

Protocol Design, Performance Evaluation, Scalability, Contention

ACM Reference Format:
Ivan Fardin, Stefano Milani, Francesca Cuomo, and Ioannis Chatzigiannakis. 2022. Enabling Edge Computing over LoRaWAN: A Device-Gateway Coordination Protocol. In *Proceedings of the 12th ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet '22), Oct. 24–28, 2022, Montreal, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3551662.3560926

1 INTRODUCTION

LoRaWAN [21] is a Low-Power Wide-Area Network (LPWAN) technology that is used to deliver highly-scalable, low-cost networks of devices integrating sensing, computation, and wireless communication that are embedded in complex physical indoor and outdoor environments. LoRaWAN follows a licence-free ad-hoc deployment model where network operators can deploy their Network Gateways (GW) in any density they require over distances that could span up to the range of several Kilometers. More often than not, this deployment freedom leads to situations of overlapping network areas covered by nearby GWs. In contrast to other LPWAN technologies, such as cellular narrowband IoT (NB-IoT), LoRaWAN allows a frame received by one or more GW to be forwarded through the network backbone to a central Network Server (NS).

Allowing multiple wireless networks to overlap simplifies the deployment process and reduces the cost of deployment. Moreover, the centralized LoRaWAN architecture helps decouple the network infrastructure from the cloud-based applications that rely on packets arriving from the IoT devices, thus further simplifying network deployment and operation. On the other hand, overlapping networks may cause unexpected packet collisions and duty-cycle exhaustion leading to deterioration of network performance. Furthermore, using GWs as simple bridges places a lot of pressure on

Part of the activities are related to the H2020 ELEGANT project (Grant agreement ID: 957286). The Authors thank the ELEGANT consortium.

DIVANet '22, October 24–28, 2022, Montreal, Canada
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9482-6/22/10…$15.00
https://doi.org/10.1145/3551662.3560926

the network backbone as the number of messages received from IoT devices can be delivered multiple times.

In this paper a novel approach is proposed that allows the IoT devices to explore the network and identify overlapping networks. A network monitoring and statistics collection mechanism is introduced so that IoT devices can assess the link quality of each overlapping network in addition to the load of the GW associated with each network. A distributed algorithm empowers each IoT device independently to choose the GW through which it will be connected to the NS. The activation process of LoRaWAN is extended by introducing a rendezvous protocol that allows devices to be associated to a specific GW that becomes the "rendezvous point" (RP) for the given device. This new layer of rendezvous points reduces the packets sent over IP from the gateways to the NS and also coordinates the downlink traffic from the NS to the devices.

Apart from improving network performance, the rendezvous protocol allows the Network Gateway to act as a point for data processing by following the Edge computing paradigm. The RP associated to a specific device participates in the agreement of a set of keys with the device. By doing so the associated RP is able to access the application payload of the LoRaWAN frames thus enabling Edge Computing. In this sense, the approach presented here transforms the GWs into active elements that become an intermediate operations layer between the IoT devices and the NS providing computational and storage resources. This approach can potentially alleviate the pressure on the network backbone if data arriving from the IoT devices are processed and aggregated at the network gateways.

The proposed protocol is implemented in software using the OMNeT++ simulator so that the performance can be evaluated in simulated environments. The evaluation builds upon the existing OMNeT++ FLoRa module for simulating LoRa interference and extends it by improving the environment's accuracy and truthfulness, considering the imperfect orthogonality of the spreading factors and the channel adjacency in LoRa [11]. Two different evaluation scenarios are used: (a) a realistic scenario based on the LoED dataset [6] to evaluate the benefit of the solution based on the operation of a real deployment during 100 days; (b) a stochastic model used to generate synthetic environment to evaluate the scalability of the proposed solution. The performance evaluation highlights the benefits of the distributed approach proposed here and provides valuable indications on the overall performance of the network.

2 RELATED WORKS AND COMPARISON

During the past years, several studies have examined the performance of LoRaWAN and how it can be improved, however very few have examined how to solve the problem of overlapping networks. One way to view this issue is during network planning as a gateway placement problem, an approach that is based on the considerable amount of results already available in the relevant literature for planning wireless and cellular networks. Unfortunately given the ad-hoc deployment model of LoRaWAN, such proposals cannot be fully implemented.

More recently, some approaches acknowledge the inability to control the placement of the GWs and thus propose a way to coordinate the operation of the network using the central role of the NS and based on the link conditions as they were reported by the GWs in terms of RSSI and SNR [15], opposed to our solution where the main actor of the protocol is the device itself. Other studies propose the use of a local controller to schedule traffic across the overlapping networks [8]. In another approach, FLIP [14], introduces a load balancing protocol assuming that the GWs are capable of communicating directly with each other. All these methods attempt to resolve the problem of network contention without attempting to reduce the network traffic resulting from the relay functionality of the GW. In addition, the assignment of IoT devices to GWs requires the periodic exchange of data regarding the network state, thus further contributing to the creation of network traffic.

The performance degradation due to overlapping LoRaWAN networks has been studied by Bhatia et al. [7] that propose a protocol for mitigating conflicts. The proposed solution reduces the number of retransmissions particularly in small-load and medium-load scenarios. The authors in [8] show that this solution allows LoRa to support a large class of control systems. Similarly, the protocol presented here mitigates the conflicts of downstream packets while also enabling edge computing at the same time.

Delbruel et al. [14] follow a different approach by forming a federation of gateways. A consensus-driven and localised algorithm is used to create the federated network. The solution presented here achieves the same result, i.e., reduces the number of packets sent over the backbone IP network, thus reducing network contention, by following a distributed approach. The devices trigger the execution of the protocol and the selection of the RP provide certain benefits regarding privacy and security.

Recently, various attempts have been made to enable edge computing over LoRaWAN deployments. Amaxilatis et al. [2] introduce an intermediate layer of gateways that allows messages to be processed. In contrast to this approach, the work presented here does not modify the existing architecture of LoRaWAN, thus guaranteeing backwards compatibility.

Jalowiczor et al. [18] propose three architectures for enabling Edge Computing over LoRaWAN. The first two consist of collapsing the GW, the NS and the Application Server on a single node at the network's edge. This solution does not need any modification of the LoRaWAN specification; however, this node represents a single point of failure. The devices can activate only with this node that stores the corresponding LoRaWAN root keys. Therefore in case of failure, the devices will not be able to link with any other GW. On the other hand, the distributed algorithm proposed here allows the device to choose any GW in the network as RP, thus switching to a new GW in case of a failure. The authors proposed a third solution based on a key negotiation between the Application Server and the GW to let the latter store the keys needed to communicate with the device. Our proposal has an opposite approach, letting directly the devices agree on a shared key with the selected GW thus totally avoiding the transmission of keys across the wireless network.

Renart et al. [26] propose a publish/subscribe overlay network to discover available computational resources at the edges of the network and allocate them to the most appropriate actor at runtime. Our solution relies on a similar idea of Rendezvous Points, however, it takes into account the limitation of LoRaWAN.

Das et al. [12] introduce the Seagull framework for distributing the sensors' data on edge based on their proximity to the source and computational power. It extends the Cowbird cloud-based framework [13] using the domain-specific language SWAN-Song [24]. In contrast to these solutions, the protocol presented here is completely decentralized with each device selecting its own RP.

3 PROTOCOL DESIGN

The protocol is triggered as soon as the device completes the *Over-The-Air-Activation (OTAA) method* with the LoRaWAN Join Server as described in the latest standard [21]. At the end of the activation, the device acquires (a) the *Application Session Key (AppSKey)* (b) the *Network Session Key (NwkSKey)*. The OTAA makes use of some hard-coded keys and 128-bit AES encryption to derive the aforementioned session keys. Recently, some alternatives have been proposed to achieve a higher level of security, e.g., by employing elliptic-curve cryptography [23]. The protocol is composed by three steps:

(1) **Device-Gateway session key establishment**: To enable secure communication, each device and the GWs that are within the transmission range agree a *Device-Gateway Session Key (DGWSKey)*.
(2) **Assessing the network quality**: Via a decentralised mechanism the devices *assess the network quality* by using a beaconing mechanism similar to the one presented in [1].
(3) **Selecting the rendevous point**: The device individually chooses a single GW among those available within its transmission range that will become its *Rendezvous Point* (RP). The RP assumes the role of coordinating uplink and downlink traffic between the device and the NS as well as enabling the local processing and storage of messages following the edge computing paradigm.

3.1 Device-Gateway session key establishment

After the completion of the activation process, the device waits a random period of time before it starts the first step of the protocol that will generate the Device-Gateway Session Key (*DGWSKey*). The random delay is used here to reduce contention when multiple devices are activated concurrently. The delay minimum/maximum intervals (`DELAY_INTERVAL`) are network parameters suitably selected by the network operator. The device in compliance with LoRaWAN device class A specification broadcasts the `GENERATE_DGW_KEY` frame to all the gateways in its area. The frame includes two nonces and a timestamp indicating the expiration date (function of the expected LoRa frame airtime).

Each GW receiving the frame verifies the validity of the frame. If expired, it ignores it; otherwise, it derives the shared session key from the nonces provided by the device and in compliance with LoRaWAN device class A specification sends back a `GENERATE_DGW_KEY` frame including an `ACK`.

To protect the Device-Gateway session key establishment protocol step from replay attacks, the mechanism to detect retransmissions defined by the LoRaWAN specification [21] is used. In particular, the mechanism uses the *Uplink Frame Counter (UpFCnt)* field in the application header of the LoRaWAN frame. This counter is used as the *Sequence Number* and is included in the `GENERATE_DGW_KEY` frame. The device can detect and ignore duplicates as well using a *Downlink Frame Counter (DownFCnt)* shared by GWs per device, in a way similar to what the NS does [4]. An efficient solution for class A LoRaWAN devices [10] consists of synchronising the downlink frame counter of the gateways with the uplink frame counter so that the former will never be greater than the latter. The NS needs to keep track of the `UpFCnt` and `DownFCnt` to receive and send frames (also MAC frames) while the algorithm is running. In order to achieve this, the `SYNC_COUNTER` frame is transmitted via the backbone IP network that synchronises the NS *UpFCnt* and *DownFCnt* for a given device.

The device in compliance with LoRaWAN device class A specification will retransmit the `GENERATE_DGW_KEY` frame until it reaches the retransmission threshold `MAX_GENERATE_REPEAT`. This is done to assure that session keys are established with the greatest possible number of GWs within its radio range. The network operator needs to suitably adjust this parameter since a high number of retransmissions increases contention in the network.

3.2 Assessing the network quality

After establishing the *DGWSKey*, the protocol proceeds with the next step, that of the assessment of the network quality. For each GW with which the device has established a *DGWSKey* and in compliance with LoRaWAN device class A specification it transmits a `STATS` frame containing a `RequestID` and its `DevEUI`. Remark that the `DevAddr`, i.e., the device's unique 32-bit network-wide identifier, is always included in the application header of the LoRaWAN frame. The GW upon receiving the `STATS` frame consults an internal table maintained in local memory that contains for each device an entry including (a) `DevAddr/DevEUI`, an array of bytes used to identify the device sending the frame; (b) `RequestID`, a byte used to avoid duplicate retransmissions of frames already analysed by the GW; (c) `AckReceived`, a boolean denoting if the device has acknowledged response of the GW to the `STATS` frame; (d) `LinkQuality`, a list containing the *Received Signal Strength Indicator (RSSI)* and the *Signal-to-Noise-and-Interference Ratios (SNIRs)* measured from the last five frames received from the device. If the table does not include any entry for a given device, or it contains an entry with a smaller `RequestID`, or the `RequestID` is equal to that included in the `STATS` frame but the `AckReceived` is set to false, the GW will update the table to reflect the new frame received, otherwise, the frame will be disregarded and no further action is taken.

After the local table is updated, the GW in compliance with LoRaWAN device class A specification replies to the device by transmitting a `STATS_REPLY` frame that includes the `LinkQuality` list. Like in the previous step, for each `STATS` frame received the GW sends a `SYNC_COUNTER` frame to the NS via the backbone IP network to update the `UpFCnt` and `DownFCnt` counters.

The device upon receiving a `STATS_REPLY` frame, estimates the RSSI and SNIR to assess the downstream network quality with the specific GW and uses the values included in the `LinkQuality` list to assess the upstream network quality. Then in compliance with LoRaWAN device class A specification it broadcasts a `STATS_ACK` frame. The GW upon receiving the frame updates the respective entry in the table by setting `AckReceived` to true and sends a `SYNC_COUNTER` frame to the NS via the backbone IP network.

Since multiple devices may be concurrently at the step of assessing the network quality, this may lead to some frames being lost and thus a sub-optimal collection of statistics. The network operator needs to carefully adjust the retransmission parameters to guarantee that enough responses are collected to acquire an accurate assessment of the network quality while at the same time not hindering the overall performance of the network due to multiple retransmissions. The number of retransmissions depends on the number of devices deployed, their geographical distribution and external factors that could cause any interference. To comply with LoRaWAN device class A, the frames should be retransmitted a fixed number of times plus the possible times for ACKs.

A new run of the algorithm can occur in two scenarios: (a) the device triggers a new OTAA (for example, after a reboot); however, in this case, the NS will assign a new *DevAddr* to the device, so the new execution of the algorithm will be independent of the previous ones; (b) to reassess the upstream and downstream network quality. Since the *DevAddr* does not change, after 256 different runs, there will be a repetition of the `RequestID`. When the *RequestID* reaches the value of 255, the device shall trigger a new OTAA to refresh its *DevAddr* before executing a new algorithm run with `RequestID` equals to `0`. The latter solution since the gateway will always drop all frames with `RequestID` less than or equal to the value stored in its table. Refreshing the `DevAddr` enables a significant resistance against replay attacks and avoid a straightforward mapping between the `DevEUI` and the `DevAddr` [3, 29].

3.3 Selecting the rendezvous point

After collecting the network quality metrics, the device identifies the GW with the most stable quality using techniques similar to those presented in [9], using the *RSSI* and the *SNIR* collected in the previous step. The device initialises the association process by broadcasting the `PAIRING_REQUEST` frame in compliance with LoRaWAN device class A specification and encrypted using the corresponding *DGWKey* including the GW address. When the designated GW receives the frame it acknowledges it by responding with a `PAIRING_ACCEPT` frrame. It also transmits a `CONNECTION_GATEWAY` frame to the NS, including the device `DevAddr`/`DevEUI`. The device upon the reception of the `PAIRING_ACCEPT` frame from the `RP` sends a `CONNECTION` frame to the `NS`, including the GW address.

Since the NS could be the entry point of the final user's requests, it maintains a *table of associations*, which is crucial to give it a complete view of the network and redirect those requests to the proper GW to enable advanced services like those described in [5]. Moreover, the `CONNECTION_GATEWAY` and `CONNECTION` frames are used by the NS to mitigate the possibility of fake associations from malicious devices and GWs. If the table of associations does not include a valid entry for the given Device-GW, the NS will acknowledge the new association, otherwise, it will ignore the request.

In particular, the acknowledgement of the new association will trigger the establishment of a new shared key, between the Device, the GW and the NS. The NS sends the `GENERATE_ASSOCIATION_KEY` frame to the GW containing the IP address of the GW and a nonce. The GW generates a nonce and derives the *shared session key* upon receiving the frame. Then, the GW replies to the device with a `GENERATE_ASSOCIATION_KEY` containing the two nonces. The device receives the nonces and computes the *same shared session key*. At this point, the association is completed, and the device will send to this RP all data frames.

4 PROTOCOL ANALYSIS

In this section, the correctness of the Device-Gateway coordination protocol is examined along with the guaranteed security properties. In addition, the message complexity of the protocol is measured in terms of the expected number of frames exchanged.

4.1 Correctness

The protocol is executed as soon as the device completes the activation process of the LoRaWAN. Therefore the case where the device is not within the transmission range of any GW is excluded. Thus, we assume that the device is within the transmission range of at least one GW. During the execution of the three steps of the protocol, all transmissions comply with LoRaWAN device class A specification. Therefore during the first step of the protocol, as presented in Sec. 3.1, the device will establish a *DGWKey* with at least one GW. During the second step of the protocol, as presented in Sec. 3.2, we assume that the device will manage to exchange at least one set of `STATS`, `STATS_REPLY` and `STATS_ACK` frames. Thus in the third step of the protocol, as presented in Sec. 3.3, the device will choose one GW as its Rendevous Point.

4.2 Security Properties

The execution of the three steps of the protocol guarantees the confidentiality and the integrity of the information exchanges through the use of different shared symmetric keys.

In the first step, the device and the GWs within its transmission range agree on the *DGWSKey* shared key that is used to encrypt the exchanged LoRaWAN frames exchanged.

The key is ephemeral since it is used only during the second step of the algorithm and discarded once the device chooses its RP. At each algorithm execution, the device and the GWs generate a new shared key, so a refresh mechanism for the *DGWSKey* is not required.

During the third step, once the device has chosen its RP, the device, the GW and the NS compute a new symmetric key that is used to encrypt all the subsequent uplink and downlink LoRa frames. A mechanism to refresh the latter key is not defined, so the device shall start a new protocol execution to refresh the cryptographic material. Besides, the device updates the GW statistics, and it chooses a new RP if convenient.

4.3 Frame Transmissions

In this section, we examine the performance of the protocol in terms of message exchanges given the protocol presented in Sec. 3. During the first phase we have 1 uplink frame (`GENERATE_DGW_KEY`), and 1 downlink for each GW. In the second phase the device sends a `STATS` to each GW in its radio range, and a `STATS_ACK` for each `STATS_REPLY` received. During the last phase the device sends a `PAIRING_REQUEST` and a `CONNECTION` frame, while the downlink frames are a `PAIRING_ACCEPT` and a `GENERATE_ASSOCIATION_KEY`. Let `r` be the retransmission parameter and `g` the number of GW in the radio range of a device the uplink and downlink frames in the worst case scenario are compute as in Eq 1 and Eq. 2 respectively.

$$\mathcal{N}_{ul} = (r)+(g+r+g+r)+(r+r) = (r)+2(r+g)+2r = 5r+2g \quad (1)$$

$$\mathcal{N}_{dl} = (g) + (g) + (r + r) = 2g + 2r \quad (2)$$

5 EXPERIMENTAL EVALUATION

The proposed coordination protocol is implemented using the *OMNeT++* [30] software simulation environment so that the performance can be evaluated further. The *OMNeT++* is an extensible, modular, component-based C++ discrete event simulation environment. The simulation of LoRa networks is supported through the *FLoRa* (Framework for LoRa) [28] that includes modules that simulate the LoRa physical layer, the LoRaWAN MAC protocol as well as network elements including gateways and network servers.

In any performed evaluation, we set the `MAX_GENERATE_REPEAT` to 3 and the `DELAY_INTERVAL` to [0, 600].

5.1 Towards a realistic interference model

In this work we conduct certain extensions to *FLoRa* to further improve the truthfulness of the interferences in large-scale and dense urban and metropolitan areas. For this reason the communication range is fixed to 5 km and the interference range to 10 km to model potential overlaps between two radio transmissions [17, 27].

According to the LoRaWAN regional parameters specification [20], the devices compute the appropriate *Spreading Factor* (*SF*) to communicate with the gateways based on their corresponding distance. The selected *SF* dictate the *Coding Rate* (*CR*, from 1 to 4) and the *Bandwidth* (*BW*). The *Data Rate* (*R*) is derived based on Eq. 3.

$$R = SF \frac{\frac{4}{4+CR}}{\frac{2^{SF}}{BW}} 1000 \quad (3)$$

When a device or a GW sends a frame, it computes the expected *Time on Air* (*T*) using Eq. 4.

$$T = T_{preamble} + T_{payload} \quad (4)$$

$T_{payload}$ is computed using Eq. 5.

$$T_{payload} = n_{payload}\ T_s \quad (5)$$

The number of symbols in the payload ($n_{payload}$) are obtained using Eq. 6.

$$n_{payload} = 8+ \max\left\{\left\lceil \frac{8L - 4SF + 28 + 16 - 20H}{4(SF - 2DE)} \right\rceil (CR+4), 0\right\}; \quad (6)$$

Here 8 is the number of symbols of the LoRa physical header that is summed up with the number of symbols of the LoRa physical frame. *L* is the length of the datalink frame; *SF* is the spreading factor; *H* is 0 if the header is explicit, 1 otherwise; *DE* is 1 if low date rate optimisation is enabled and *CR* is the coding rate (from 1 to 4) [19, 20].

The duration of a single symbol (T_s) is obtained from Eq. 7.

$$T_s = \frac{2^{SF}}{BW} \quad (7)$$

On the other hand, $T_{preamble}$ is computed using Eq. 8

$$T_{preamble} = \left(n_{preamble} + n_{syncword}\right)\ T_s \quad (8)$$

where $n_{preamble}$, $n_{syncword}$ and $n_{payload}$ are the number of symbols in the preamble, in the synchronization word and in the payload of a single frame.

Every LoRaWAN device maintains a set of devices that are within its radio range and identifies all possible intersections of the radio ranges of nearby devices. These intersections of pair of devices are used to identify possible collisions of concurent frame transmissions. Notice that since the position of the devices in a simulated scenario are static, this computation is performed only at the start of an experiment. For each transmitted frame, using the above calculated *Time on Air* any frame transmitted by pairs that fall within the identified intersections are examined for possible interference.

For each pair of identified frames that may have possible interference, the *physical parameters* of the frames are retrieved (*RSSI, spreading factor, bandwidth, frequency channel*) as well as the *Signal-to-Noise-and-Interference Ratios (SNIRs)* to compute the external interference [25, 31]. If the spreading factors, bandwidth and channel frequencies of the frames match, the *Signal Interference Ratio (SIR)* between the interferer and the interfered frames and their preambles is computed [16]. On the contrary, the interference could be weak because of the imperfect orthogonality of the spreading factors. In this case, two thresholds are set for the SIR and the *Bit Error Rate (BER)* according to the spreading factor. To estimate the impact of the interference, if the SIR exceeds the threshold, we compute the BER based on an experimental curve [11]. When two messages are transmitted in different channels, the interference may occur if the interferer is transmitted during the preamble transmission of the interfered and the spreading factor used by the former is at least 10. Based on the channels adjacency, the probability of packet loss is estimated [22].

5.2 Stochastic model

The first set of experiments looks into the scalability of the proposed solution in terms of the total number of deployed devices and GWs. The goal of these experiments is to measure the number of messages exchanged, both uplink and downlink until all deployed devices select a single GW. These evaluation scenarios do not attempt to reflect realistic deployments but worst-case scenaria regarding the possible interferences. The compliance with the duty cycle defined by the LoRaWAN specifications is enforced by using a simple timeout mechanism. When an uplink is transmitted, the end device updates the consumed LoRaWAN duty cycle and starts a timeout to reset it. The end device will not proceed with another transmission before the duty cycle is exhausted.

The deployment area is set to a rectangular $5km \times 5km$ and (a) the first set of experiments is generated by deploying a fixed number of GW (3) and deploying a varying number of devices (10 . . . 320); (b) the second set of experiment is generated by deploying a fixed number of devices (100) and deploying a varying number of GWs (2 . . . 16). In both cases, the GWs are placed at equal distances among them while the devices are placed randomly in the area.

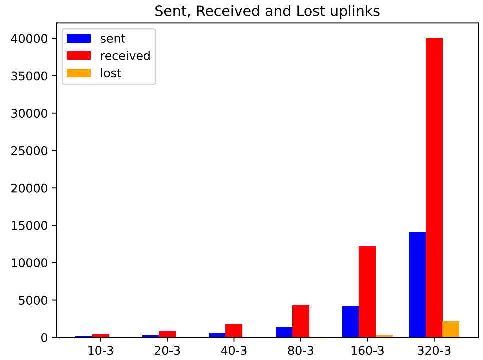

Figure 1: LoRa uplink frames metrics. 3 **gateways and** 10, 20, 40, 80, 160, 320 **devices**

During each experiment, the number of uplink and downlink message exchanges conducted during the execution of the coordination protocol are counted and also the number of messages dropped due to an identified collision based on the realistic interference model presented in the previous section. So, for example, if a device sends a frame and two gateways receive it while one does not, the uplink will be counted twice in the received metric and once in the lost metric. Each experiment is repeated multiple times until the observed variables lie between the bounds of the 95% confidence interval for 95% of the experiments.

Fixed Gateways. The goal of these experiments is to estimate the overhead produced by the coordination protocol as the number of devices increases. Fig. 1 depicts the number of uplinks LoRa frames sent by the devices, received by the gateways, and the lost ones. As expected, increasing the number of devices all the metrics grow. However, the impact on the number of frame losses is minimal.

Fig. 2 depicts the number of downlink LoRa frames sent by the gateways, received by the devices, and the lost ones. Again, all metrics are expected to grow as the number of devices increases. However, contrarily to the uplink case, lost downlink frames grow much faster, with a strong impact on the efficiency of the proposed protocol when more than 80 devices are deployed in the radio ranges of 3 gateways. Nevertheless, the protocol always succeeds in assigning each device to a single GW despite the increased interference in dense deployments.

Fixed Devices. The next set of experiments looks into the overhead produced by the coordination protocol as the number of overlapping networks increases. Looking into Fig. 3 that depicts the statistics for the uplink LoRa frames we notice, similar to the previous case, an increasing number of lost frames as the number of gateways increases. On the other hand, examining the downlink frames based on Fig. 4, the LoRa network performance drops exponentially, increasing the number of gateways. From 4 gateways and on, the number of lost frames grows, however, the protocol always succeeds in assigning each and all of the devices to a single GW despite the increased interference in overlapping networks.

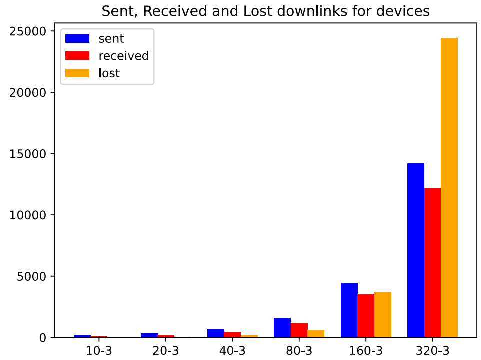

Figure 2: LoRa downlink frames metrics. 3 **gateways and** 10, 20, 40, 80, 160, 320 **devices**

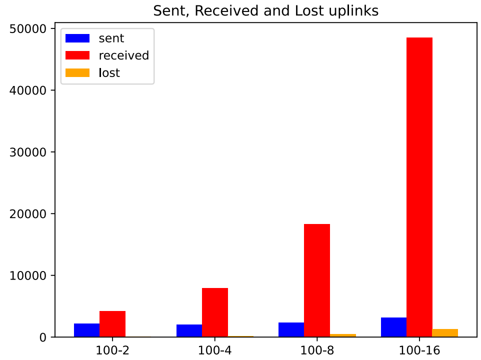

Figure 3: LoRa uplink frames metrics. 100 **devices and** 2, 4, 8, 16 **gateways**

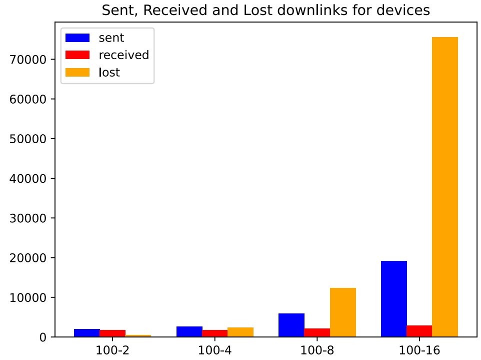

Figure 4: LoRa downlink frames metrics. 100 **devices and** 2, 4, 8, 16 **gateways**

The experiments indicate that in dense deployment scenarios where GWs have overlapping areas of network coverage the transmission of downlink messages is heavily affected. The deployment

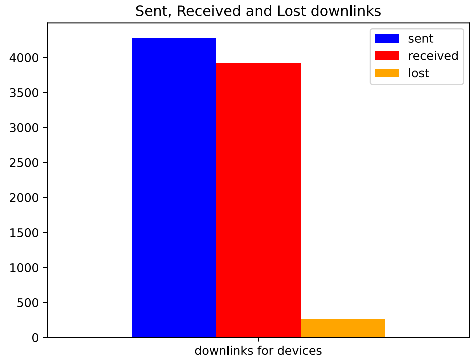

Figure 5: Number of sent, received and lost data downlink LoRa frames in the realistic scenario

scenaria of 8 or 16 overlapping GWs constitutes worst-case deployments. Although the protocol always succeeds to complete the assignment despite the interference, it is critical to properly select the position of the gateways to avoid too many radio overlappings. Alternatively, the downlink frame losses can be mitigated by increasing the random period that the devices will wait before starting the execution of the protocol.

5.3 Realistic model

The previous experiments evaluated the scalability of the protocol by measuring only the number of control frames exchanged. In this section, the evaluation looks into the exchange of data frames exchanged in a real application. The LoED dataset [6] is used to evaluate the benefit of the solution based on the operation of a real deployment for 100 days. In particular, a total of 8 GWs and 173 end devices are deployed over the metropolitan area of London. During this period all the frames received are recorded along with all the physical parameters. Now apart from the frames exchanged during the initialization of the proposed framework, the data frames transmitted are also included in the statistics. Moreover, additional statistics are collected regarding the traffic generated while the GW forward the received uplink frames to the NS via the backbone IP network.

The number of downlinks lost frames is depicted in Fig. 5. In contrast to the worst-case network scenaria evaluated in the previous section, in this real-world network deployment, the number of lost frames is limited. Interestingly, as depicted in Fig. 6, the number of lost uplink frames is zero.

After the execution of the coordination protocol, all the end devices successfully select a single GW. In this way, all data frames are forwarded to the NS only once. As a result, there is a significant reduction of messages sent over the IP backbone network compared to the actual LoRaWAN specification, where all gateways receiving an uplink frame should forward it to the NS. Comparing Fig. 6 and Fig. 7, the number of data uplink received by the gateways are about 1.6×10^6 while the data frames forwarded to the NS via the IP backbone network are about 0.8×10^6 since only the chosen RP forward the processed data LoRa frame to the NS.

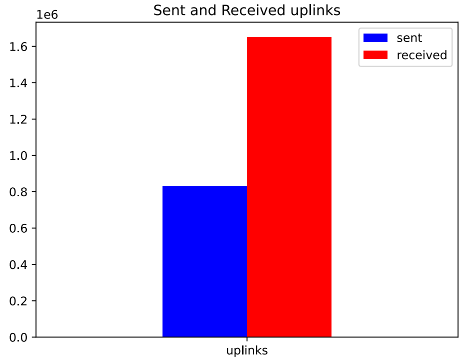

Figure 6: Number of sent and received data uplink LoRa frames in the realistic scenario

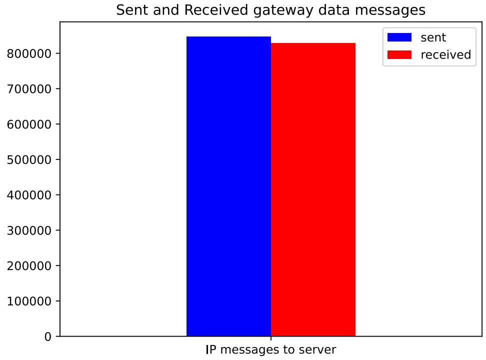

Figure 7: Number of sent, received data IP frames to the Network Server in the realistic scenario

6 CONCLUSION

The work presented here examines the impact of the LoRaWAN ad-hoc deployment model on network scalability in cases of overlapping areas of network coverage. A novel device-gateway coordination protocol is presented that enables each device individually to identify overlapping networks and select in a decentralized way a single GW to act as a rendezvous point. This assignment helps reduce the interference across both uplink and downlink messages and also reduces the traffic over the backbone IP network connecting the GWs with the NS. Moreover, it can be used to implement the edge computing paradigm over LoRaWAN deployments.

The performance of the protocol is studied in terms of message overhead created, correctness in terms of successfully assigning each device to a single GW and the security properties. A series of experiments are conducted to investigate the scalability of the protocol and study the performance improvements when employed in real-world deployments. The accuracy of the OMNeT++ simulation environment for modelling interference in dense deployments is improved by implementing a set of extensions to consider the non-perfect orthogonality of the LoRa Spreading Factors and the channel adjacency. The experimental analysis of the extended OMNeT++ environment indicates that a large number of overlapping networks have a significant impact on the number of collisions on downlink frames during the execution of the protocol. Nevertheless, the protocol succeeds in assigning each device to a single GW. The experimentation in real-world scenarios also confirms that the downlink frames are the most critical issue in terms of performance, however, it also shows the positive impact of the coordination protocol presented here on the overall performance of the network, reducing almost by half the total number of frames forwarded to the LoRaWAN Network Server via the backbone IP network.

Several future research directions are worth investigating. First, we wish to look into the possibility of allowing the gateways to assist the devices in selecting a rendezvous point by introducing an initial gateway-to-gateway coordination phase. The improvement aims to reduce the total number of downlink messages. Of course, the decentralized approach presented here that allows each device to select individually the gateway that it wishes to be assigned has certain benefits in terms of privacy and security. It is, therefore, crucial to guarantee that the involvement of the gateways in the selection of the rendezvous points will maintain the security properties of the protocol. Another direction is to improve the protocol in terms of channel interference with the assigned rendezvous point in which case the coordination protocol needs to be restarted. Here it is worth investigating the use of a Group Key among all the gateways that are within the transmission range of a single device. When the assigned gateway is temporarily unreachable, the device can easily be re-assigned to another gateway without repeating the entire protocol. Another direction is the evaluation and the consequent improvement of the novel interference model in more extensive networks deployed in larger spaces.

REFERENCES

[1] Dimitrios Amaxilatis and Ioannis Chatzigiannakis. 2018. Design and Analysis of Adaptive Hierarchical Low-Power Long-Range Networks. *Journal of Sensor and Actuator Networks* 7, 4 (2018). https://doi.org/10.3390/jsan7040051

[2] Dimitrios Amaxilatis, Ioannis Chatzigiannakis, Christos Tselios, Nikolaos Tsironis, Nikos Niakas, and Simos Papadogeorgos. 2020. A Smart Water Metering Deployment Based on the Fog Computing Paradigm. *Applied Sciences* 10 (3 2020), 1965. Issue 6. https://doi.org/10.3390/app10061965

[3] Lucas Ancian and Mathieu Cunche. 2020. Re-identifying addresses in LoRaWAN networks. (9 2020). https://hal.inria.fr/hal-02926894https://hal.inria.fr/hal-02926894/document

[4] Emekcan Aras, Gowri Sankar Ramachandran, Piers Lawrence, and Danny Hughes. 2017. Exploring the security vulnerabilities of LoRa. *2017 3rd IEEE International Conference on Cybernetics, CYBCONF 2017 - Proceedings* (7 2017). https://doi.org/10.1109/CYBCONF.2017.7985777

[5] Ahmet Cihat Baktir, Atay Ozgovde, and Cem Ersoy. 2017. Enabling service-centric networks for cloudlets using SDN. *Proceedings of the IM 2017 - 2017 IFIP/IEEE International Symposium on Integrated Network and Service Management* (7 2017), 344–352. https://doi.org/10.23919/INM.2017.7987297

[6] Laksh Bhatia, Michael Breza, Ramona Marfievici, and Julie A. McCann. 2020. Dataset: LoED: The LoRaWAN at the Edge Dataset. *DATA 2020 - Proceedings of the 3rd Workshop on Data Acquisition To Analysis, Part of SenSys 2020, BuildSys 2020* (10 2020), 7–8. https://doi.org/10.1145/3419016.3431491

[7] Laksh Bhatia, Po-Yu Chen, Michael Breza, Cong Zhao, and Julie A McCann. 2021. IRONWAN: Increasing Reliability of Overlapping Networks in LoRaWAN. *IEEE Internet of Things Journal* (2021).

[8] Laksh Bhatia, Ivana Tomić, Anqi Fu, Michael Breza, and Julie A. Mccann. 2021. Control Communication Co-Design for Wide Area Cyber-Physical Systems. *ACM Trans. Cyber-Phys. Syst.* 5, 2, Article 18 (jan 2021), 27 pages. https://doi.org/10.1145/3418528

[9] Konstantinos Chantzis, Dimitrios Amaxilatis, Ioannis Chatzigiannakis, and José Rolim. 2014. Symmetric coherent link degree, adaptive throughput-transmission power for wireless sensor networks. In *2014 IEEE International Conference on Distributed Computing in Sensor Systems*. IEEE, 26–34.

[10] Phui San Cheong, Johan Bergs, Chris Hawinkel, and Jeroen Famaey. 2017. Comparison of LoRaWAN classes and their power consumption. *2017 IEEE Symposium on Communications and Vehicular Technology, SCVT 2017* 2017-December (12 2017), 1–6. https://doi.org/10.1109/SCVT.2017.8240313

[11] Daniele Croce, Michele Gucciardo, Stefano Mangione, Giuseppe Santaromita, and Ilenia Tinnirello. 2018. Impact of LoRa Imperfect Orthogonality: Analysis of Link-Level Performance. *IEEE Communications Letters* 22 (4 2018), 796–799. Issue 4. https://doi.org/10.1109/LCOMM.2018.2797057

[12] Roshan Bharath Das, Gabriele Di Bernardo, and Henri Bal. 2018. Large Scale Stream Analytics Using a Resource-Constrained Edge. *2018 IEEE International Conference on Edge Computing (EDGE)*, 135–139. https://doi.org/10.1109/EDGE.2018.00027

[13] Roshan Bharath Das, Nicolae Vladimir Bozdog, and Henri Bal. 2017. Cowbird: A Flexible Cloud-Based Framework for Combining Smartphone Sensors and IoT. *2017 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, 1–8. https://doi.org/10.1109/MobileCloud.2017.14

[14] Stéphane Delbruel, Nicolas Small, Emekcan Aras, Jonathan Oostvogels, and Danny Hughes. 2020. Tackling Contention Through Cooperation: A Distributed Federation in LoRaWAN Space. In *Proceedings of the 2020 International Conference on Embedded Wireless Systems and Networks on Proceedings of the 2020 International Conference on Embedded Wireless Systems and Networks* (Lyon, France) *(EWSN '20)*. Junction Publishing, USA, 13–24.

[15] Domenico Garlisi, Ilenia Tinnirello, Giuseppe Bianchi, and Francesca Cuomo. 2021. Capture Aware Sequential Waterfilling for LoRaWAN Adaptive Data Rate. *IEEE Trans. Wirel. Commun.* 20, 3 (2021), 2019–2033. https://doi.org/10.1109/TWC.2020.3038638

[16] Jetmir Haxhibeqiri, Floris Van den Abeele, Ingrid Moerman, and Jeroen Hoebeke. 2017. LoRa Scalability: A Simulation Model Based on Interference Measurements. *Sensors 2017, Vol. 17, Page 1193* 17 (5 2017), 1193. Issue 6. https://doi.org/10.3390/S17061193

[17] Jetmir Haxhibeqiri, Eli De Poorter, Ingrid Moerman, and Jeroen Hoebeke. 2018. A Survey of LoRaWAN for IoT: From Technology to Application. *Sensors 2018, Vol. 18, Page 3995* 18 (11 2018), 3995. Issue 11. https://doi.org/10.3390/S18113995

[18] Jakub Jalowiczor, Jan Rozhon, and Miroslav Voznak. 2021. Study of the Efficiency of Fog Computing in an Optimized LoRaWAN Cloud Architecture. *Sensors 2021, Vol. 21, Page 3159* 21 (5 2021), 3159. Issue 9. https://doi.org/10.3390/S21093159

[19] Ji-Young Jung and Jung-Ryun Lee. 2021. Throughput and Packet Loss Probability Analysis of Long Range Wide Area Network. *Applied Sciences* 11, 17 (2021). https://doi.org/10.3390/app11178091

[20] LoRa Alliance. 2017. *LoRaWAN 1.1 regional parameters.* Technical Report.

[21] LoRa Alliance. 2017. *LoRaWAN 1.1 Specification.* Technical Report.

[22] Jinshan Luo, Atsushi Ito, Akira Sasaki, Madoka Hasegawa, Yoshikazu Nagao, Yuko Hiramatsu, Kotaro Torii, Shiori Ashibe, and Toru Aoki. 2020. A Study on Adjacent Interference of LoRa. In *2020 Eighth International Symposium on Computing and Networking Workshops (CANDARW)*. 35–39. https://doi.org/10.1109/CANDARW51189.2020.00020

[23] S. Milani and I. Chatzigiannakis. 2021. Design, analysis, and experimental evaluation of a new Secure Rejoin Mechanism for LoRaWAN using Elliptic-Curve Cryptography. *Journal of Sensor and Actuator Networks* 10 (2021). Issue 2. https://doi.org/10.3390/JSAN10020036

[24] Nicholas Palmer, Roelof Kemp, Thilo Kielmann, and Henri Bal. 2012. SWAN-Song: a Flexible Context Expression Language for Smartphones. (2012).

[25] Ladislav Polak and Jiří Miloš. 2020. Performance analysis of LoRa in the 2.4 GHz ISM band: coexistence issues with Wi-Fi. *Telecommunication Systems* 74 (07 2020). https://doi.org/10.1007/s11235-020-00658-w

[26] Eduard Gibert Renart, Javier Diaz-Montes, and Manish Parashar. 2017. Data-Driven Stream Processing at the Edge. *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, 31–40. https://doi.org/10.1109/ICFEC.2017.18

[27] Semtech. [n.d.]. LoRa and LoRaWAN: Technical overview. https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan

[28] Mariusz Slabicki, Gopika Premsankar, and Mario Di Francesco. 2018. Adaptive configuration of lora networks for dense IoT deployments. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*. 1–9. https://doi.org/10.1109/NOMS.2018.8406255

[29] Pietro Spadaccino, Domenico Garlisi, Francesca Cuomo, Giorgio Pillon, and Patrizio Pisani. 2021. Discovery privacy threats via device de-anonymization in LoRaWAN. (8 2021), 1–8. https://doi.org/10.1109/MEDCOMNET52149.2021.9501247

[30] Andras Varga. 2010. *OMNeT++*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-12331-3_3

[31] Benny Vejlgaard, Mads Lauridsen, Huan Nguyen, Istvan Z. Kovacs, Preben Mogensen, and Mads Sorensen. 2017. Interference Impact on Coverage and Capacity for Low Power Wide Area IoT Networks. In *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. 1–6. https://doi.org/10.1109/WCNC.2017.7925510