A Testbed for Time-Slotted LoRa Communications

Dimitrios Zorbas
Tyndall National Institute, University College Cork, Ireland
Email: dimzorbas@ieee.org

***Abstract*—Industrial Internet of Things applications require wireless networking protocols that can achieve high reliability, high scalability, and allow node mobility. To this end, this demo presents a testbed for TS-LoRa; a time-slotted and collision-free protocol for LoRa-based networks for applications that require frequent and extremely reliable transmissions. The demo consists of a gateway system as well as several end-nodes that can join the network, synchronise with it, and autonomously get a unique frame slot to perform transmissions. In this demo, a general description of the system along with its processes is provided.**

I. Introduction

TS-LoRa [1] has been proposed as a LoRa-based candidate to support Industrial Internet of Things (IIoT) applications. These applications require a guaranteed latency, a high packet delivery ratio, and a low power consumption. To achieve these requirements, TS-LoRa uses time-slotted and collision-free transmissions. It adopts a novel mechanism to send acknowledgements and achieve periodic synchronisation using a single downlink packet. It also relies on the traditional LoRaWAN architecture where each node can reach a gateway using a single hop.

Firstly, in contrast to other radio technologies in IIoT which rely on multi-hop deployments, LoRa can achieve a much longer range, thus, intermediate relays that increase the deployment cost are not required anymore. Apart from that, the extended coverage of LoRa increases the freedom of mobility within the network and eliminates the need of re-computing costly multi-hop routes. Secondly, LoRaWAN networks (the current open standard for LoRa devices) suffer from extensive collisions due to the Aloha-based MAC layer. This means that some typical IIoT requirements mentioned above cannot be guaranteed using LoRaWAN.

Using time-slotted LoRa communications as with TS-LoRa, the collisions can be eliminated, thus, high scalability and reliability can be achieved. However, this does not come without pitfalls. For example, LoRa transmissions in sub-GHz ISM bands are restricted to duty cycle regulations and it is not easy to organise the nodes in slots due to their often different LoRa radio characteristics [2]. TS-LoRa tackles most of these issues while exhibiting a better energy consumption than LoRaWAN for frequent packet transmission [1].

In this demo, a testbed for time-slotted LoRa transmissions is presented. This testbed implements TS-LoRa and shows insights of its functionalities.

II. LoRa and Radio Duty Cycle Restrictions

LoRa is a proprietary spread spectrum modulation (SSM) and long range radio technology currently owned by Semtech [3]. LoRa's main feature is that it can trade data rate with sensitivity and, thus, achieve a longer range at the expense of a lower data rate. To do so, it makes use of a radio parameter, called Spreading Factor (SF) which adjusts the spread in the SSM. Signals with higher SF values can be detected with significantly lower sensitivity resulting in link budgets of over 150dBm. However, the higher the SF, the lower the data rate. This means that for the same payload, a node needs to spend more energy as the SF increases because the transmission lasts longer. The user can also configure LoRa transmissions with a set of other parameters, such as the channel bandwidth, the coding rate, the preamble size, and the cyclic redundancy check which can also affect the transmission time. Some of these parameters are used to increase the error detection and correction levels of the transmissions. On the receiver side, the user can choose between single-channel and multi-channel gateways. In the first case, the receiver can listen to only one channel and SF at a time, while in the second case, up to 8 parallel transmissions can be decoded. Moreover, parallel transmissions performed on different SFs can be simultaneously decoded by a multi-channel gateway.

LoRa mainly uses license-free sub-GHz radio frequency bands (e.g., EU868, US915) that are restricted to radio duty cycle regulations. In Europe, ETSI has reserved a number of ISM frequency bands, where most of the bands have a 1% duty cycle limit and maximum 25mW (14dBm) Effective Isotropic Radiated Power (EIRP) for transmissions [4]. The duty cycle restriction imposes delays between successive transmissions which in turn limits the capabilities in scheduling transmissions and transmitting acknowledgements [2].

III. TS-LoRa

TS-LoRa exploits several parallel and repeated frames (one for each SF). The number of slots in the frame depends on the application requirements, the number of nodes, and the regional radio duty cycle rules. An additional slot is used at the end of each frame for synchronisation and acknowledgements (SACK slot). Guard times (empty time) are added in between slots to tolerate clock desynchronisations, and thus, to avoid overlaps between successive transmissions. Fig. 1 illustrates two parallel TS-LoRa frames for the first two SFs and a typical frame structure consisting of n slots. We must note that TS-LoRa respects the EU radio duty cycle rules, so the nodes are allowed to transmit at 1% of the total time. The current implementation of TS-LoRa, supports delay flexible applications, so additional slots can be added to support more

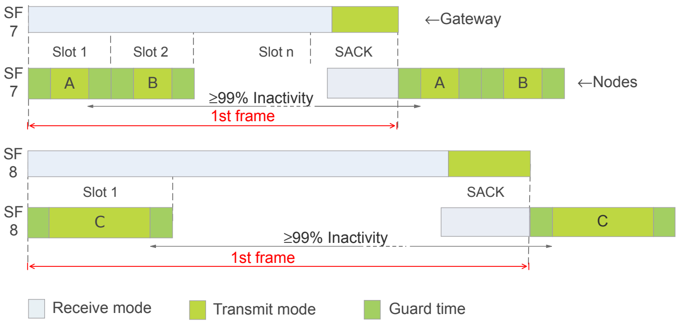


Fig. 1. TS-LoRa repeated and parallel frames accommodating two (A, B) and one (C) node for the first two Spreading Factors respectively.

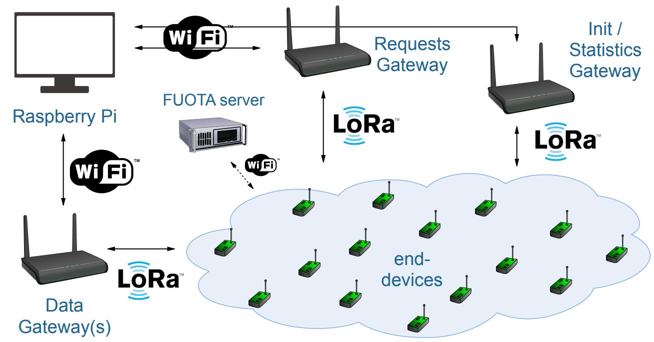


Fig. 2. The testbed architecture.

nodes at the expense of a higher delay. However, the code can be easily adapted to delay constrained applications [5]. In this case, the frame size is fixed as the delay requirements dictate, however, a limited number of nodes can be supported.

IV. Architecture, Demonstration & Interaction

A. Architecture & Components

In this demo, TS-LoRa is implemented as a stand-alone protocol using Pycom Lopy4 nodes (https://pycom.io/) and the Micropython programming language (https://github.com/deltazita/ts-lora/).

The architecture consists of multiple gateways, a Raspberry Pi, and the end-nodes, as it is depicted in Fig. 2. One of the gateways is used to handle the join requests, while one or more gateways are used for the data collection and the transmission of the acknowledgements. For the sake of simplicity and in order to independently send acknowledgements per SF, current TS-LoRa implementation uses multiple 1-channel gateways – one for each SF. The implementation also includes a gateway for experiment initialisation and statistics collection once an experiment is finished. The nodes can send out information regarding the number of re-transmissions, the dropped packets, and the acknowledged packets.

In the present testbed, the Raspberry Pi plays the role of the network and application server, thus, it is only used to host the key generation mechanism during the registration. All the gateways have two network interfaces, the IEEE802.11n to communicate with the Raspberry Pi and the LoRa interface. The gateways and the Raspberry Pi are located close to each other and their communication is done over a secure channel. The system also supports a Firmware Upgrade Over The Air (FUOTA) mechanism using WiFi. FUOTA is very useful to run experiments with many nodes as it does not require an one-by-one firmware or code update.

B. Experiment flowchart

Fig. 3 presents the flow of the processes that a node follows once the user initiates a TS-LoRa experiment. An experiment is initiated by sending a special init packet to the Raspberry Pi TCP server. The init packet contains the number of packets that need to be transmitted by the nodes as well as information about the FUOTA server. The Raspberry Pi forwards this information to one of the gateways over WiFi which is then forwarded to the nodes over LoRa. For the needs of the experiment, the nodes must have their radio on and wait for an init packet. Once the init packet is received, the nodes can optionally communicate with the local WiFi network to perform a firmware update. The next step is the node registration. Registration requests are sent over SF12 while the Channel Activity Detection (CAD) mechanism of LoRa transceivers may be used to reduce the probability of collisions. TS-LoRa uses a similar to LoRaWAN OTAA mechanism (Over The Air Activation) where an application key (AppKey) and the device identifier (DevEUI) are used during the registration (see Fig. 4). For convenience, current TS-LoRa implementation uses node ids as well, however, the node ids are not involved in the registration mechanism and how the security keys are generated. Once the registration is successful, the node proceeds with the slot calculation which is extracted from the device address (DevAddr) given to the node from the network server during the registration. This is an autonomous slot generation mechanism which is part of the TS-LoRa registration [1]. A node can then receive the first sync packet which is broadcasted periodically by the data gateway(s). Once a node is synchronised with the gateway's clock, it can compute the wake-up time for the next transmission as well as the wake-up time to receive the first SACK packet. This is a loop process which finishes once the maximum number of successfully delivered packets is reached. The experiment finishes when all the nodes send their statistics to the statistics collection gateway.

C. Demonstration & Interaction

The demo will provide insights of the system architecture including the gateways' functionalities, the autonomous slot generation mechanism, the node registration, and the data collection process. The audience will have the opportunity to see in detail the processes described in the previous subsection and the packets exchanged in these processes using the node's standard output, as it is depicted in Fig. 5. More specifically, the user can see when a new round (frame) started, the synchronisation packet, and how many packets were acknowledged and re-transmitted. The nodes and the gateways can additionally inform the user about the background processes

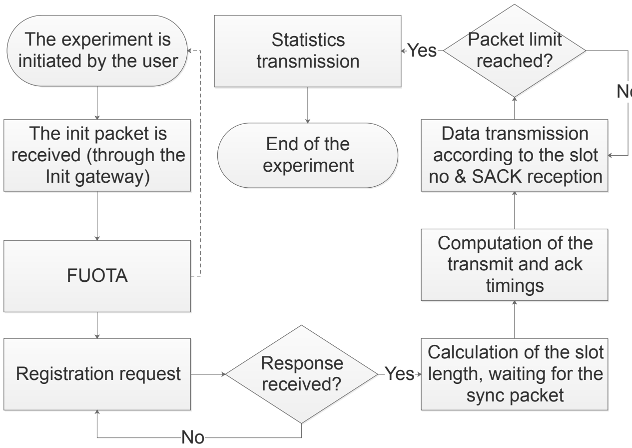


Fig. 3. The flowchart of processes of an end-node.

```
Pycom id: b4e62df1dc15 My id: 35
LoRa DevEUI: 70b3d5499d04909c
Waiting for commands...
OTA over WLAN (IP): 192.168.1.6
Performing OTA
Connected to WiFi!
Requesting: manifest.json?current_ver=1.0.15
Already on the latest version
Request sent! b'#%70b3d5499d04909c:3efd4267ef71836a:1:7'
Device: 1 - Pkg:  b'35:f5ef4c90:5'
Slot = 0 DevAddr =  f5ef4c90
Waiting for sync...
ACK!
time after SACK rec (ms): 16.1499
sync slot lasted (ms): 1916.473
joining the network lasted (ms): 5241.502
-----
airtime (ms): 41.216
Guard time (ms): 15.0
Duty cycle slots: 58
SACK slot length (ms): 80.0
Gw processing time (ms): 8
S T A R T
```

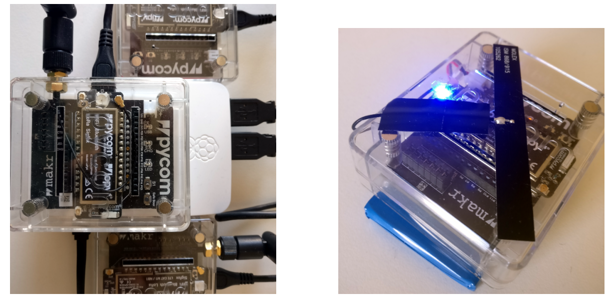
Fig. 4. Standard output during initialisation, registration, & synchronisation.

described in the previous subsection using flashing LEDs. More specifically different colours are used for the registration, synchronisation, and data transmission. The gateway system and an end-node are displayed in Fig. 6.

The demonstration will also show results as those presented in Fig. 7. These results are provided by 7 nodes after a 6-hour experiment and 5000 transmitted packets per node. We can see the number of unique transmitted packets, the delivered ones, the number of re-transmissions, and the number of dropped packets. In this particular example, a 99.98% packet delivery ratio was achieved.

```
5 -----------------------------------------------------
starting a new round at (ms): 0.0108
sleep time (ms): 3.0
Sending packet of 12 bytes at (ms): 14.421
sleep time after data (s): 4.08444 / clock correction (ms): 0.0
started sync slot at (ms): 4154.006
SACK received! 4:8:f
(rx_timestamp=66943469, rssi=-63, snr=6.0, sfrx=7, sftx=7, tx_trials=0
ACK!
new sync slot length (ms): 74.914
sync slot lasted (ms): 74.01575
time after SACK (ms): 13.624
transmitted/delivered/retransmitted/dropped: 5 5 0 0
round lasted (s): 4.236974
radio active time (rx/tx) (s): 0.3700411 / 0.2820498
```

Fig. 5. Node's standard output during a round (frame).



(a) The TS-LoRa gateway system

(b) A TS-LoRa end-node with a dipole antenna

Fig. 6. TS-LoRa gateways and an end-node.

```
Got connection from ('192.168.0.254', 35170)
Stopped accepting stats!
Ready to accept stats!
Node 21: 5000 5000 44 0
Node 34: 5000 4998 47 2
Node 19: 5000 5000 77 0
Node 22: 5000 4999 96 1
Node 35: 5000 5000 139 0
Node 15: 5000 5000 177 0
Node 14: 5000 4998 203 2
```

Fig. 7. Statistics received after a 6h experiment with 7 nodes (Node id: packets transmitted delivered retransmitted dropped).

V. Conclusions

In this demo, a testbed for time-slotted LoRa communications was presented. TS-LoRa, a time-slotted communications protocol for LoRa-enabled devices, is implemented on the testbed. The required hardware as well as the flowchart of the processes were described. Indicative results from a set of 7 nodes were finally shown.

Acknowledgements

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) and is co-funded under the European Regional Development Fund under Grant Number 13/RC/2077. It has also received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 713567.

References

[1] D. Zorbas, K. Abdelfadeel, P. Kotzanikolaou, and D. Pesch, "TS-LoRa: Time-slotted LoRaWAN for the Industrial Internet of Things," *Computer Communications*, vol. 153, pp. 1 – 10, 2020.

[2] D. Zorbas, "Design Considerations for Time-Slotted LoRa(WAN)," in *Proceeding of the International Conference on Embedded Wireless Systems and Networks (EWSN)*, pp. 1–6, ACM, Feb 2020.

[3] Semtech Corporation, "LoRa Modulation Basics." https://lora-developers.semtech.com/library/product-documents/, 2015. Online; accessed 11-May-2020.

[4] ETSI EN 300 220-2, "Short Range Devices (SRD) operating in the frequency range 25 MHz to 1 000 MHz; Part 2: Harmonised Standard for access to radio spectrum for non specific radio equipment," 2018.

[5] D. Zorbas, "Improving Delay and Capacity of TS-LoRa with Flexible Guard Times," in *21st IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pp. 1–6, IEEE, 2020.