

Lab3: LoRaWAN and TTN

Marco Zennaro, PhD
ICTP



Labs

- 1/3 Ready to use, tested examples
- 1/3 Exercise based on the examples
- 1/3 Your imagination → create new applications

Our Lab equipment

Pycom LoPy 4

PySense

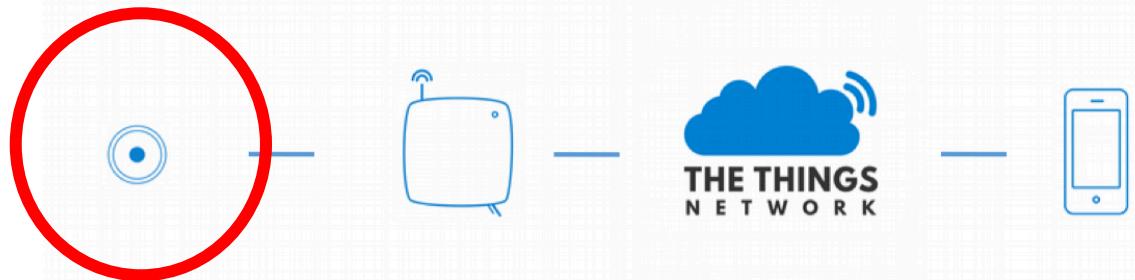
microUSB Cable

LoRaWAN Gateway



TTN: devices, gateways, servers

HOW DOES THIS WORK?



DEVICES

GATEWAYS

NETWORK
SERVER

APPLICATION
SERVER



Sending T,H to TTN

TTN: App

As a first step we must create a TTN application and register our device to it. This is necessary so that data are correctly encrypted.

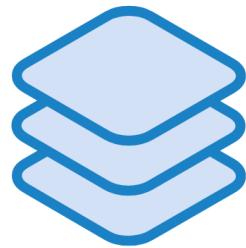
Create a new application in TTN.

TTN: App

👋 Hi, Marco!

Welcome to The Things Network Console.

This is where the magic happens. Here you can work with your data. Register applications, devices and gateways, manage your integrations, collaborators and settings.



APPLICATIONS



GATEWAYS

TTN: App

Application ID

ADD APPLICATION

Application ID
The unique identifier of your application
The Things Network

`test_application_fablab`

Description
A human readable description of your new application
Eg. My sensor network application

Application EUI
An application EUI will be issued for The Things Network block for convenience. You can add your own in the application settings
Generated by The Things Network

Handler registration
Select the handler you want to register your application to
`ttn-handler-eu`

Description

Handler (Europe)

TTN: we have a new App!

APPLICATION OVERVIEW

Application ID **test_application_fablab**

Description

Created 11 seconds ago

Handler **ttn-handler-eu** (*current handler*)

APPLICATION EUIS

70 B3 D5 7E D0 01 70 74

TTN: Collaborators

DEVICES

 0 registered devices

[register device](#) [manage devices](#)

COLLABORATORS

 marcozennaro

[manage collaborators](#)

[collaborators](#) [delete](#) [devices](#) [settings](#)

TTN: add a Collaborator to the App

ADD COLLABORATOR

Could not add application
An app with the application id test_3 already exists.

Username

Erm| 

 Ermanno Ermanno Pietrosemoli

Rights

settings
Manage the application settings and access keys

collaborators
Edit the application collaborators

delete
Delete the application

devices
View and edit devices of the application

TTN: register a device

REGISTER DEVICE

Device ID
This is the unique identifier for the device in the app. The device ID will be generated.

Device EUI
The device EUI is the unique identifier for this device on the network. You can change the EUI later.

App Key
The App Key will be used to secure the communication between your device and the network.
 this field will be generated

App EUI
 70 B3 D5 7E D0 01 70 74

Name of Device
Device EUI

Where is the device EUI?

Step 1: Create a device in TTN with the OTAA keys from LGT-92.

Each LGT-92 is shipped with a sticker with the default device EUI as below:



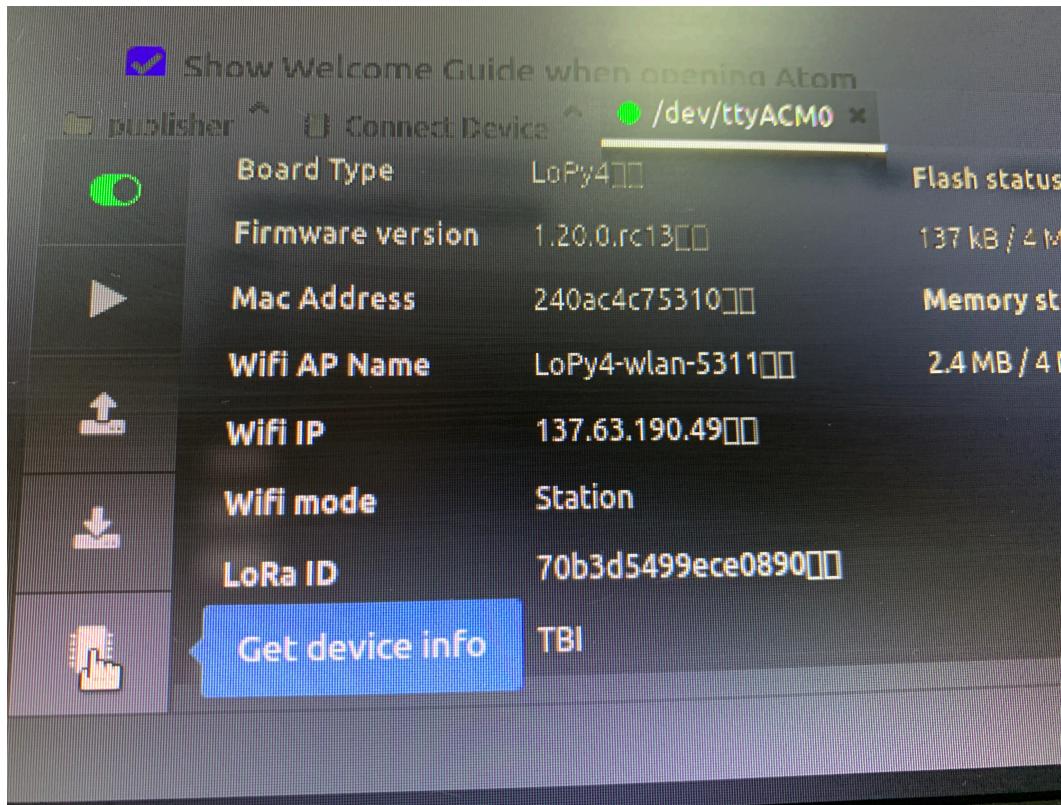
Device EUI for LoPy

To obtain the Device EUI of your LoPy, execute the following code in your REPL console:

```
from network import LoRa  
import binascii  
lora = LoRa(mode=LoRa.LORAWAN)  
print(binascii.hexlify(lora.mac()).upper().decode('utf-8'))
```

As an output you will receive a string that contains the Device EUI.

Device EUI for LoPy



TTN: devices

REGISTER DEVICE

Device ID
This is the unique identifier for the device in this app. The device ID will be immutable.

Device EUI
The device EUI is the unique identifier for this device on the network. You can change the EUI later.



App Key
The App Key will be used to secure the communication between your device and the network.

 this field will be generated

App EUI

TTN: devices

DEVICE OVERVIEW

Application ID test_application_fablab

Device ID test_device

Activation Method OTAA

Device EUI <> 70 B3 D5 49 95 AB DB CE

Application EUI <> 70 B3 D5 7E D0 01 70 74

App Key <> ...

Status • never seen

Frames up 0 [reset frame counters](#)

Frames down 0

Authentication

Never seen!

TTN: devices

Settings

A screenshot of the TTN Device Overview page. At the top right, there is a navigation bar with three tabs: "Overview" (highlighted in blue), "Data", and "Settings". A large red arrow points downwards towards the "Settings" tab. The main section is titled "DEVICE OVERVIEW". It contains the following fields:

- Application ID:** test_application_fablab
- Device ID:** test_device
- Activation Method:** OTAA
- Device EUI:** 70 B3 D5 49 95 AB DB CE
- Application EUI:** 70 B3 D5 7E D0 01 70 74
- App Key:** (redacted)
- Status:** never seen

TTN: devices

SETTINGS

Description
A human-readable description of the device

Device EUI
The serial number of your radio module, similar to a MAC address
 70 B3 D5 49 95 AB DB CE 8 bytes

Application EUI
 70 B3 D5 7E D0 01 70 74

Activation Method
 OTAA ABP



ABP

TTN: devices

Activation Method

OTAA

ABP

Device Address

The device address will be assigned by the network server

Network Session Key



Network Session Key will be generated

App Session Key



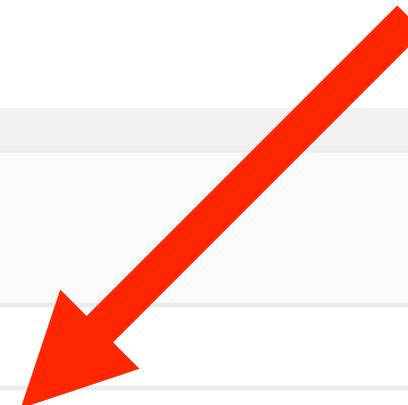
App Session Key will be generated

TTN: devices

DeviceAdd, NetKey, AppKey

EXAMPLE CODE

```
1 const char *devAddr = "26011607";
2 const char *nwkSKey = "09827AA1D4BBDB382859F47A49F6C20B";
3 const char *appSKey = "6B54FDB99BF4A1E90A768C3B5FAD3F50";
```



TTN App: first example

Open the example in the Code/LoRa/TTN directory.

This example code sends a short message "1,2,3" to TTN using ABP authentication.

TTN App: first example

```
dev_addr = struct.unpack(">I",  
    binascii.unhexlify('260118A2'))[0]
```

Modify these values with the ones provided by
TTN for your application

```
nwk_swkey =  
binascii.unhexlify('F913FB6F4E47  
169234163839D5A76787')
```

```
app_swkey =  
binascii.unhexlify('CB4DECE3104  
D7B5EB85AFFD8334E45E3')
```

TTN App: first example

In TTN's Application, you should now be able to see
the data coming in.



TTN App: first example

The screenshot shows a web browser displaying the TTN Console at <https://console.thethingsnetwork.org>. The URL in the address bar is https://console.thethingsnetwork.org/applications/testuganda/devices/device_test/data. The page title is "THE THINGS NETWORK CONSOLE COMMUNITY EDITION". The navigation path is Applications > testuganda > Devices > device_test > Data. On the right, there are tabs for Overview, Data (which is selected and highlighted with a red circle), and Settings. Below the tabs, there is a control bar with pause and clear buttons. The main content area is titled "APPLICATION DATA" and contains a table with the following data:

time	counter	port	payload
▲ 15:26:23	6	2	01 02 03
▲ 15:26:18	5	2	01 02 03
▲ 15:26:13	4	2	01 02 03
▲ 15:26:08	3	2	01 02 03
▲ 15:26:03	2	2	01 02 03
▲ 15:25:58	1	2	01 02 03

TTN App: T,H

Open the example in the
Code/LoRa/TTN+Pysense/pycom directory.

This example code reads T and H from the Pysense
and sends this information via TTN.



TTN App: T,H example

If your devices are transmitting data properly, all messages received will be seen in TTN.

To check the incoming messages from the devices, go to the "Traffic" tab from gateway console.

TTN: payload

Payload format

The screenshot shows the TTN application overview page. At the top, there is a navigation bar with tabs: Overview (highlighted in blue), Devices, Payload Formats (highlighted with a red arrow), Integrations, Data, and Settings. Below the navigation bar, the section title "APPLICATION OVERVIEW" is displayed in blue. To the right of the title is a link to "documentation". The main content area contains the following information:

- Application ID:** test_application_fablab (highlighted with an orange box)
- Description:** (empty)
- Created:** 30 minutes ago
- Handler:** ttu-handler-eu (*current handler*)

TTN: payload

PAYOUT FORMATS

Payload Format
The payload format sent by your devices

Custom

decoder converter validator encoder

```
1 function Decoder(bytes, port) {  
2   // Decode an uplink message from a buffer  
3   // (array) of bytes to an object of fields.  
4   var decoded = {};  
5  
6   // if (port === 1) decoded.Led = bytes[0];  
7  
8   return decoded;  
9 }
```

TTN: payload

Open the payload example in the
Code/LoRa/TTN+Pysense/ttn-decoder directory.

Copy the decoder as payload decoder in TTN.



TTN App: T,H example

On TTN you should now be able to see the data coming in and you should be able to decode the payload (so you can read Temperature and Humidity).

T,H TTN: Exercises

- 1) Move in the lab and check the RSSI values as seen by TTN. How far can you go?
- 2) Create one Application for the whole class and add ALL devices to it.

Summary

We learned how to send data to TTN.

We visualized data using the Ubidots integration.

Feedback?

Email mzennaro@ictp.it