TP LP-WAN – 3SN – Novembre/Décembre 2020 – Riadh DHAOU

Initiation à LoRaWAN

Objectifs :

1. Installation et configuration d'un réseau LoraWAN composés de deux passerelles : Indoor GW et Outdoor GW (sur le toit du bâtiment C de l'ENSEEIHT)
2. Observation du trafic généré et analyse des résultats expérimentaux en fonction des paramètres LoRa.

0) Installer la passerelle indoor LoRaWAN

Visualiser à l'aide de wireshark le trafic échangé entre la GW et les serveurs privé ACKLIO, ou publique The Things Network (TTN).

1) Installer les outils de programmation des devices (nodes) LoRA

```
sudo su
```

Installer **nodejs**

```
curl -sL https://deb.nodesource.com/setup_13.x | sudo -E bash -

apt-get update

apt-get install nodejs
```

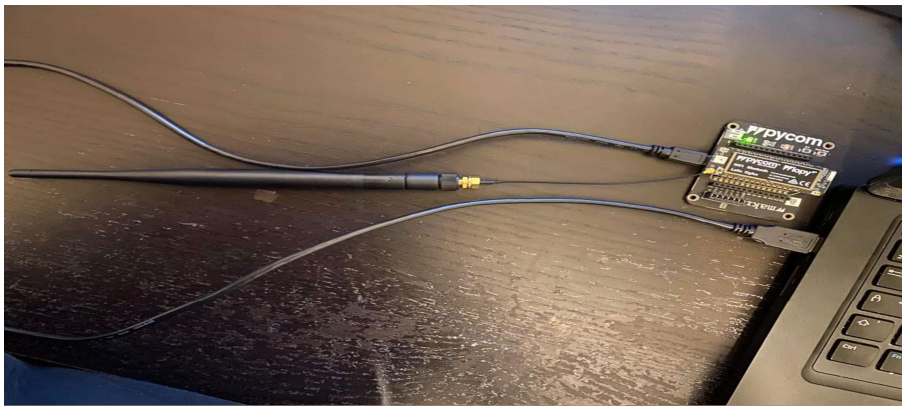
Récupérer le paquetage de VSC (le .deb) sur https://code.visualstudio.com/Download

Installer VSC à l'aide de

```
sudo dpkg -i ~/Downloads/code_*.deb; sudo apt -f install
```

Pour aller plus loin : plus de détail sur la programmation des devices se trouve sur le site voir site https://docs.pycom.io/gettingstarted/connection/lopy4/ (en particulier le Tutorial et les exemples).

Faites attention à l'emplacement des broches et de l'antenne (voir la partie matériel du tutorial pour plus de précisions)



2) Programme de base pour récupérer l'adresse du device

Récupérer l'identifiant unique des nodes (le DevEUI).

3) Activation des devices

Configurer la passerelle en mode serveur puis ajouter votre LopyDev à la liste des Devices autorisés de la passerelle.

Se connecter au serveur TTN https://console.thethingsnetwork.org (Login/Passwd : duiot/duiot2020).

Vous pourrez créer vos applications et ajouter des Devices ou d'autres GWs (tous les Devices ont été ajoutés sur une application par défaut). Ce serveur permet de générer les informations nécessaires à l'authentification (Application EUI, et App Key).

Activation selon le mode Over The Air Activation (OTAA) des devices sur les deux serveurs : sur la GW (pour le mode Serveur), et sur le serveur d'Acklio (pour le mode Packet Forwarder) [1]

Dans ce TP, le mode d'activation adopté est le mode OTAA. Pourquoi avoir fait ce choix ? Quelle est la différence par rapport au mode ABP ?

Vous aurez besoin pour cette activation des informations suivantes : DevEUI, AppKey, AppEUI (expliquer comment ces clés sont générées). Compléter la fiche de présence avec ces informations.

Le serveur www.loratools.nl apportera des outils utiles pour la génération des clés et pour le calcul des paramètres de temps de vol (Air Time) en fonction des paramètres LoRa (Coding Rate, Spreading Factor, bande passante,…)

4) Programmer les nodes

Créer un premier programme qui permet d'envoyer les données à la GW selon le mode de classe A

[1] Le serveur d'Acklio https://enseeiht.cores.acklio.net/ fourni plus de fonctionnalités au prix d'un abonnement. D'autres serveurs sont très largement utilisés comme Orange LiveObjects.

Créer un projet sous VSC avec l'exemple LORA de base et le mode OTAA (voir exemple sur le tutorial de Lopy4)

Choisir les paramètres des nodes (LORAWAN, EU868, APP EUI, APPKEY, mode classe A…)

Ce programme doit permettre l'envoi des requêtes (JOIN) puis envoyer les données à la GW.

Analyser le trafic à l'aide de Wireshark (entre GW et serveur)

Observer les données reçues sur le trafic du serveur TTN[2]

Expliquer le fonctionnement pour un device de classe A.

Observer le trafic reçu directement sur la GW (le login est admin, l'adresse IP est 192.168.2.3, le mot de passe est entré par l'enseignant). Cette observation n'est possible que si la passerelle est en mode serveur.

```
nc  - -udp   - - listen   - - local-port 1784
```

Les données générées par votre Device sont-elles reçues? Si oui, par quelle passerelle ?

5) Réaliser une étude expérimentale

L'objectif est d'étudier l'impact des paramètres LoRA (datarate, Spreading Factor (SF) et Coding Rate (CR) sur les performances de la communication.

On s'attachera particulièrement à observer la qualité du signal reçu, le temps mis sur le support en fonction du couple datarate/Spreading Factor.

Quel est l'impact du paramètre SF ? (Vous pouvez illustrer au travers de graphiques en traçant les résultats des expérimentations en jouant sur ce paramètre mais aussi sur la charge du trafic généré (en adaptant la durée entre deux transmissions).

Des mesures peuvent être faites sur la GW ou sur l'état de trafic reçu sur le serveur TTN[3] peuvent permettre de mesurer les paramètres physiques lors de la réception.

Ces paramètres peuvent également être récupérés au niveau des Devices.

---

[2] (ou sur l'Observatory du serveur Aklio)

[3] ou des observations sur l'Observatory d'acklio

Exemples de codes utiles (voir tutorial pour aller plus loin)

- pour la récupération du Dev EUI

```
from network import LoRa
import pycom
import ubinascii
#
lora = LoRa(mode=LoRa.LORAWAN, region = LoRa.EU868)
mac = lora.mac()
#
print ('devEUI: ',  end='')
#
for i in range (0,  8):
      print(hex(mac[i]), end='-')
#
print ()
print ("DevEUI: %s" % (ubinascii.hexlify(lora.mac()).decode('ascii')))
```

- pour l'envoie de données sur le réseau LoRa

```
from network import LoRa
import socket
import time
import ubinascii
# Initialise LoRa in LORAWAN mode.
# Please pick the region that matches where you are using the device:
# Asia = LoRa.AS923
# Australia = LoRa.AU915
# Europe = LoRa.EU868
# United States = LoRa.US915
lora = LoRa(mode=LoRa.LORAWAN, region=LoRa.EU868)
# create an OTAA authentication parameters, change them to the provided credentials
app_eui = ubinascii.unhexlify('70b3d57ed0038811')
app_key = ubinascii.unhexlify('05BDA4F60A4A0AAFEBDD7320ECCB9F70')
#uncomment to use LoRaWAN application provided dev_eui
dev_eui = ubinascii.unhexlify('70b3d54999f3a539')
# join a network using OTAA (Over the Air Activation)
#uncomment below to use LoRaWAN application provided dev_eui
#lora.join(activation=LoRa.OTAA, auth=(app_eui, app_key), timeout=0)
lora.join(activation=LoRa.OTAA, auth=(dev_eui, app_eui, app_key), timeout=0)
```

```
# wait until the module has joined the network
while not lora.has_joined():
time.sleep(2.5)
print('Not yet joined...')
print('Joined')
# create a LoRa socket
s = socket.socket(socket.AF_LORA, socket.SOCK_RAW)
# set the LoRaWAN data rate
s.setsockopt(socket.SOL_LORA, socket.SO_DR, 5)
# make the socket blocking
# (waits for the data to be sent and for the 2 receive windows to expire)
s.setblocking(True)
# send some data
s.send(bytes([0x01, 0x02, 0x03]))
# make the socket non-blocking
# (because if there's no data received it will block forever...)
s.setblocking(False)
# get any data received (if any...)
data = s.recv(64)
print(data)
```