

Manuel de l'utilisateur v1

Noeud + Passerelle LoRa, Projet Beliot



Par
Vincent Gosselin

Fait le 8 Août 2017

Table des matières

Installation de la passerelle.....	p.3-5
Installation du noeud.....	p.5-6
La Plateforme Web.....	p.7
Où trouver le travail réalisé.....	p.7
Considérations importantes.....	p.8

Installation de la passerelle

1. Brancher le Raspberry Pi dans un routeur à l'aide d'un câble Ethernet.
2. Brancher l'alimentation du Raspberry Pi.
3. Mettre le Raspberry Pi a "ON" (alimenter)
4. Découvrir et noter le IP local du Raspberry Pi à l'aide de la page web du routeur.
Chez moi, le routeur à l'adresse "http://home/".
5. À l'aide de la console tapez :
"ssh pi@XXX.XXX.XXX.XXX"
où XXX.XXX.XXX.XXX correspond au IP découvert à l'étape 4.
6. Le mot de passe est :
"raspberrypi"
7. Une fois entrer dans le raspberrypi, tapez la commande : "tmux ls".
La commande va retourner un message d'erreur si il n'y a pas de session tmux déjà lancé:

```
Vincent-Gosselins-MacBook:~ vincentgosselin$ ssh pi@192.168.3.4
pi@192.168.3.4's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Aug  1 20:24:36 2017 from 192.168.3.1

SSH is enabled and the default password for the 'pi' user has not been changed
This is a security risk - please login as the 'pi' user and type 'passwd' to
set a new password.

pi@raspberrypi:~ $ tmux ls
failed to connect to server: No such file or directory
pi@raspberrypi:~ $
```

Si il y avait un serveur tmux de disponible, il suffit de taper "tmux attach" pour se connecter à la session tmux.

8. Tapez la commande "tmux" pour démarrer une nouvelle session tmux.
Une fois la session tmux lancé, une barre verte apparait en bas d'écran.

```
pi@raspberrypi:~ $
```

```
[0] 0:bash* "raspberrypi" 20:32 01-Aug-17
```

9. Entrer la commande "cd /home/pi/Desktop/LowCostLoRaGw/lora_gateway" pour accéder au répertoire de la passerelle LoRa.

À titre d'information :

- ```
pi@raspberrypi:~/Desktop/LowCostLoRaGw/Lora_gateway $ ls
3GBongle gateway_id.txt post_to_influxdb_v3.rb
aas-python-lib HOWTo.txt post_to_webserver.py
arduPi.cpp HOWTo_V2.txt radio.makefile
arduPi.h key_FireBase.py rapidjson
arduPi.o key_GroveStreams.py README-aes_lorawan.md
arduPi_pi2.cpp key_SMS.py README-downlink.md
arduPi_pi2.h key_ThingSpeak.py README.md
arduPi_pi2.o libSMS.py README-NewCloud.md
bcm2835.h libSMS.pyc rfcomm-server.py
CloudFireBaseAES.py log_gw.py scripts
CloudFireBaseLWAAES.py lora_gateway sensors_in_raspi
CloudFireBase.py lora_gateway.cpp STARTGATEWAY.sh
CloudGroveStreams.py lora_gateway.o start_gw.py
CloudMongoDB.py lora_gateway_pi2 SX1272.cpp
clouds.json lora_gateway_pi2.o SX1272.h
CloudSMS.py loraWAN.py SX1272.o
clouds_parser.py makefile SX1272_pi2.o
clouds_parser.pyc MongoDB.py test-loraWAN-1.py
CloudThingSpeak.py php test-loraWAN-2.py
cmd.sh post_processing_gw.py test-loraWAN-3.py
downlink post_to_influxdb.rb
gateway_conf.json post_to_influxdb_v2.rb
```

11. Taper "bash STARTGATEWAY.sh" pour activer la passerelle. L'écran devrait alors avoir l'allure suivante :

```
Current working directory: /home/pi/Desktop/LowCostLoRaGw/lora_gateway
SX1276 detected, starting.
SX1276 LF/HF calibration
...
*****Power ON: state 0
Default sync word: 0x12
LoRa mode 1
Setting mode: state 0
Channel CH_5_900: state 0
Set LoRa power dBm to 14
Power: state 0
Get Preamble Length: state 0
Preamble Length: 8
LoRa addr 1: state 0
SX1272/76 configured as LR-BS. Waiting RF input for transparent RF-serial bridge
Low-level gw status ON

[0] 0:~$
```

```
[0] 0:bash* "raspberrypi" 20:58 01-Aug-17
```

```
pi@raspberrypi:~$ tmux ls
failed to connect to server: No such file or directory
pi@raspberrypi:~$ tmux
[detached (from session 0)]
pi@raspberrypi:~$
```



13. Taper "exit" pour se déconnecter du raspberry pi.

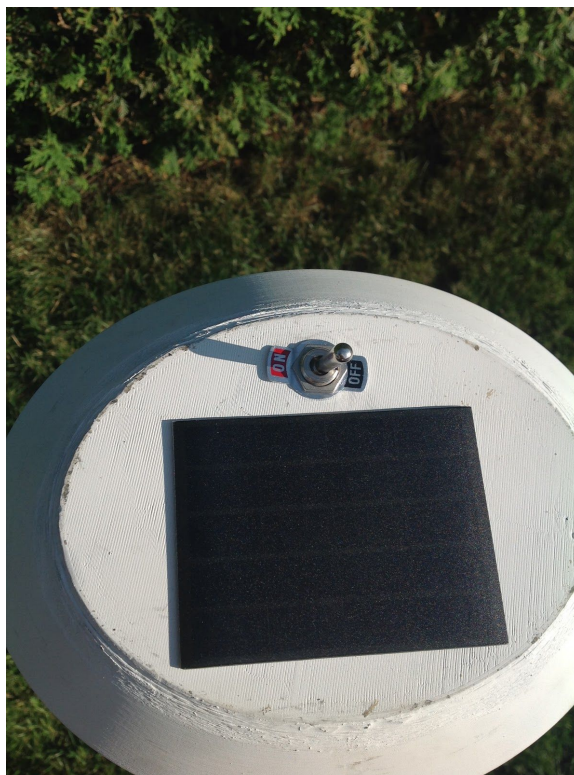
La passerelle est maintenant configuré.

## Installation de noeud

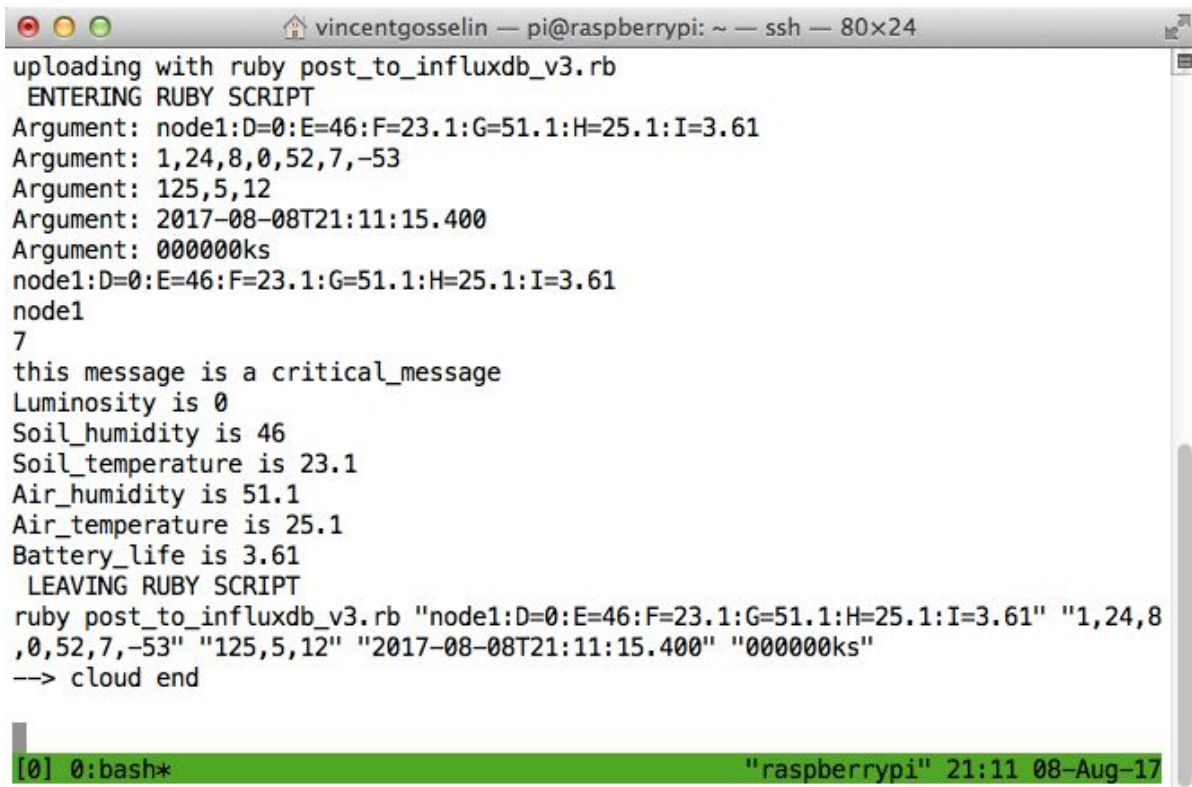
1. Planter la barre dans le sol a l'aide d'un maillet.
2. Installer la tête du prototype sur la barre de fer et visser la tête sur la barre de fer.



3. Activer la SWITCH à ON sur la tête du prototype pour mettre le noeud en mode actif.



4. Après 40 secondes, un message sera reçu sur le rasp dans la session tmux pour indiquer une communication réussi.



```
vincentgosselin — pi@raspberrypi: ~ — ssh — 80x24
uploading with ruby post_to_influxdb_v3.rb
 ENTERING RUBY SCRIPT
Argument: node1:D=0:E=46:F=23.1:G=51.1:H=25.1:I=3.61
Argument: 1,24,8,0,52,7,-53
Argument: 125,5,12
Argument: 2017-08-08T21:11:15.400
Argument: 000000ks
node1:D=0:E=46:F=23.1:G=51.1:H=25.1:I=3.61
node1
7
this message is a critical_message
Luminosity is 0
Soil_humidity is 46
Soil_temperature is 23.1
Air_humidity is 51.1
Air_temperature is 25.1
Battery_life is 3.61
 LEAVING RUBY SCRIPT
ruby post_to_influxdb_v3.rb "node1:D=0:E=46:F=23.1:G=51.1:H=25.1:I=3.61" "1,24,8,0,52,7,-53" "125,5,12" "2017-08-08T21:11:15.400" "000000ks"
--> cloud end

[0] 0: bash* "raspberrypi" 21:11 08-Aug-17
```

6. Planter dans le sol la sonde de température et la sonde d'humidité.

7. En cas, d'échec de cette réception de donnée, Switch -> OFF, Switch -> ON, devrait réussir la communication. Parfois (rarement), il y a une erreur LoRa pendant la première transmission. Donc, éteindre et ré-allumer le prototype devrait faire.

8. Noter qu'il serait important d'avoir un recouvrement de la tête sur la partie supérieure afin de protéger la partie électronique du contact de l'eau.

9. Pour programmer le arduino attaché au prototype, il est très important de mettre la SWITCH à OFF avant de connecter le port USB.

# La plateforme Web

1. La plateforme Web est accessible à l'adresse [beliotv3.herokuapp.com](https://beliotv3.herokuapp.com)
2. Pour avoir les prévisions locales, Cliquer sur l'image de la météo.
3. Pour l'historique du noeud, Cliquer sur l'image de Grafana.
4. Sur Grafana, Usager : "client" Mot de passe : "vignesbordeaux"
5. Sélectionner le Dashboard "vignes\_bordeaux\_v1"

Pour l'info, le site web est contenu sur la plateforme Heroku et est codé en Ruby. Le code du site web est disponible sur Github : <https://github.com/vincentgosselin1/beliotv3> . La base de données Influxdb + Grafana sont hébergé sur la plateforme de Amazon Web Services.

Notez que la gratuité de la plateforme Web sur Amazon Web Services prend fin Mai 2018, prière de communiquer [vincentgosselin1@gmail.com](mailto:vincentgosselin1@gmail.com) avant la date d'échéance.

## Où trouver le travail réalisé

1. La totalité du travail se trouve à l'adresse : [https://github.com/vincentgosselin1/Arduino\\_node](https://github.com/vincentgosselin1/Arduino_node)
2. Le code principale qui doit être flashé dans le arduino nano se trouve dans le dossier Main et se nomme "main.cpp". Afin de compiler le code dans le arduino, il est important de copier/coller la totalité du fichier main.cpp dans un fichier .ino pour flasher le arduino. En plus, il faut copier le contenu du dossier "libraries" dans le répertoire des libraries Arduino. Sur mon Macbook, la location de ce dossier se trouve dans "/Users/vincentgosselin/Documents/Arduino/libraries".
3. Les projets Kicad pour les circuits électriques ainsi que les pcbs sont inclus dans le Github dans la section Electronics.

## Considération importantes

1. Le module consomme plus que prévue, un erreur au niveau de la conception électrique sur un régulateur force la consommation à 150mA en mode actif et à 60mA en mode économie d'énergie. La faute est à cause de la ligne 5V qui ne va pas à 0V quand le régulateur "shut-off". En enlevant le GPS + Détecteur de poussière, la consommation passe à 40mA en mode actif et 10mA en mode économie d'énergie. Pour une raison qu'on ignore encore, le mode économie d'énergie observer sur plaque d'essais, ne descend pas jusqu'à 0.25mA. Ceci est à investiguer.
2. Il faut recouvrir la partie supérieure de la tête du prototype pour le rendre imperméable à l'eau (la partie inférieur l'est). Du simple "Duck tape" fait l'affaire.
3. Un optimisation de code a été fait mais elle s'est avéré inefficace puisque le programme atteignait 98% de mémoire statique et 80% mémoire dynamique avec l'inclusion du capteur de luminosité. La communication LoRa ne fonctionnait pas et un message de "warning" sur Arduino apparaissait. Alors, c'est pour cette raison que le capteur de luminosité ne fait pas partie du prototype. Le code est stable maintenant.