|  |  |  |
| --- | --- | --- |
| 2017 |  | **Déploiement de 2 réseaux LoRa à l’INSA** |



|  |  |
| --- | --- |
| **DOMAINE** | Formation/Recherche/Relation partenaires INSA fondation et laboratoires |
| **Chef de projet fonctionnel** | Thierry Monteil |
| **Description** | L’objectif est de déployer 2 réseaux loRa sur le campus et plus largement sur le bassin toulousain avec des connections à l’international. C’est un projet hautement stratégique pour l’INSA en terme de formation de ses ingénieurs, de visibilité de sa filière numérique à l’extérieur, de coopération avec la métropole de Toulouse, de demande de membre de la fondation INSA Toulouse et enfin de coopération en terme de recherche.   1. Réseau international TTN (The Things Network)   L’objectif est de déployer et maintenir une gateway (équipement réseau permettant de dialoguer d’une part via la technologie réseau LoRa et de réémettre sur un réseau IP en direction d’un serveur) connectée au réseau international TTN.  Un premier prototype a été déployé par les étudiants de 4IR et repris lors d’un stage (voir Annexe 1 pour plus d’information).   1. Création d’un réseau opéré par l’INSA   L’objectif est de créer un réseau pour les objets connectés opéré par l’INSA. Ceci implique le déploiement de plusieurs gateways probablement 2 sur l’INSA, une sur le LAAS, une chez Actia, une dans la métropole de Toulouse. Ces gateways doivent communiquer avec un serveur collectant les informations des objets connectés au gateway. Ce serveur doit être accessible depuis l’extérieur à la fois pour récupérer les informations des gateways extérieures à l’INSA mais aussi pour des utilisateurs référencés dans la plate-forme mais connectés sur un réseau hors INSA.  Ceci implique le déploiement d’une VM ou d’un conteneur et l’ouverture de certains ports réseau (voir Annexe 2). Afin d’assurer un niveau de service suffisant, il faudra gérer dans le temps ce réseau ainsi que ses utilisateur en collaboration avec le DGEI.  Il est à noter qu’une réunion avec l’étudiant ayant fait un stage sur de domaine pour l’étude de faisabilité est possible rapidement avant qu’il ne soit très pris afin d’avancer et faire un transfert de compétences. |
| **Echéance attendue** | Déploiement à l’automne 2017 pour une utilisation par la formation 5ISS, 4AE et les partenaires de l’INSA en novembre |
| **Information Financière** | Fourniture des équipements de type gateway par le DGEI, accès à un serveur du CSN |

|  |  |
| --- | --- |
|  | **CADRE RESERVE CSN** |
| **Date réception** |  |
| **Pôle** |  |
| **Chef de projet technique** |  |
| **Coût financier** |  |
| **Coût humain** |  |
| **Validation** | Date C2SN |
| **Planification** |  |

**Analyse de la demande :**

**ANNEXE 1**

**Rapport TTN : installer une gateway LoRa sur le réseau INSA et paramétrage**

Introduction

L’objectif de ce rapport est de présenter un document synthétique permettant de connecter une gateway LoRa TTN sur le réseau INSA pour pouvoir ensuite connecter nos capteurs.

1. **Installation de la gateway sur le réseau INSA**

Pour ce qui est de la construction de la gateway, un très bon tutoriel est disponible à l’adresse : <https://github.com/ttn-zh/ic880a-gateway/wiki>. La liste du matériel est dressée et les étapes pour mettre le code sur le raspberry sont détaillées.

Nous allons plus nous attarder sur la partie réseau permettant de relier cette gateway au réseau INSA.

Le CSN a créé des accès spéciaux pour accéder au serveur TTN.

Les règles sont les suivantes :

* + ip : router.eu.thethings.network
  + port (pour les paquets entrants et sortants) : 1700
  + protocol: UDP
  + accès SSH pour configurer le RPi

L’INSA dispose pour l’instant de 4 gateways basées sur des RPi, 4 adresses IP ont donc été débloquées par le CSN : de 193.50.229.65 à 68.

Une fois que nous avons les adresses IP, il faut configurer le RPi pour avoir une adresse IP statique.

|  |  |
| --- | --- |
| **GW N°** | **EUI** |
| 1 | B8-27-EB-08-1B-E2 |
| 2 (sur le toit du DGEI) | B8-27-EB-C4-79-DF |
| 3 | B8-27-EB-60-E9-8D |
| 4 | B8-27-EB-F1-44-DC |

Par ailleurs, il est important de noter qu’il est préférable que la gateway soit reliée à internet en ethernet pour diminuer les risques d’interférence entre le wifi et le LoRa et donc diminuer le bruit.

1. **Configuration de la gateway**

Les 4 gateways disposent d’un EUI (identifiant unique permettant de connecter la gateway sur un réseau), voici un résumé :

Les RPI n’ont qu’un compte utilisateur :

* Utilisateur : ttn
* Mdp : ttn

S’il n’y a pas de connexion ethernet, le RPi va se connecter en wifi sur le réseau suivant :

SSID: ttn

PWD: ttn12345

La configuration du réseau INSA est la suivante :

Network: 193.50.229.64/28

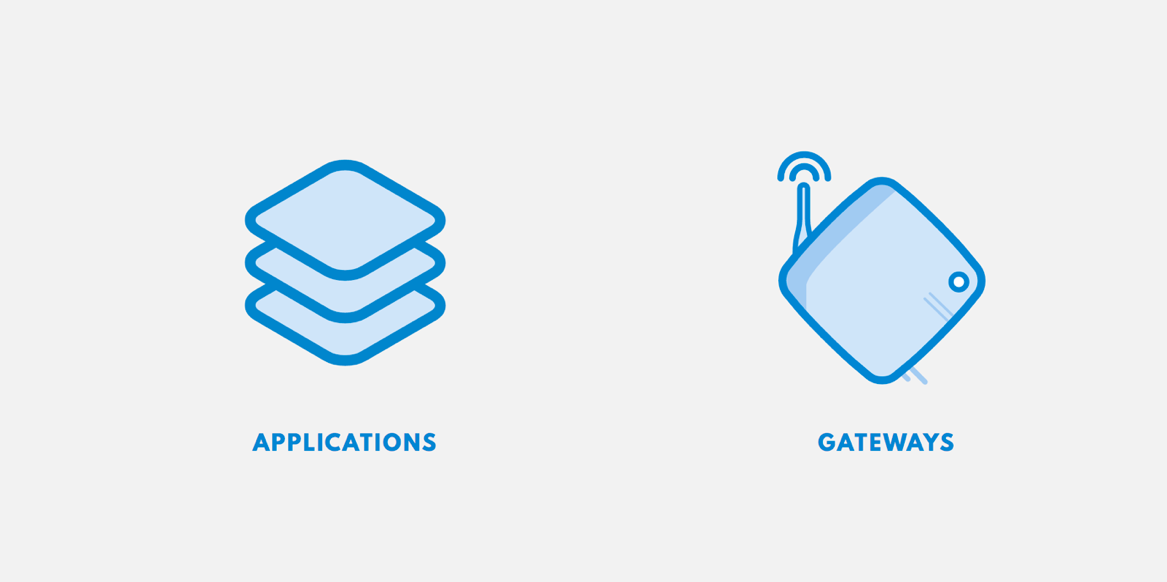
IPs: 193.50.229.65 to 68,

Gateway: 193.50.229.78

Mask: 255.255.255.240

1. **Connexion de la gateway sur le Réseau TTN**

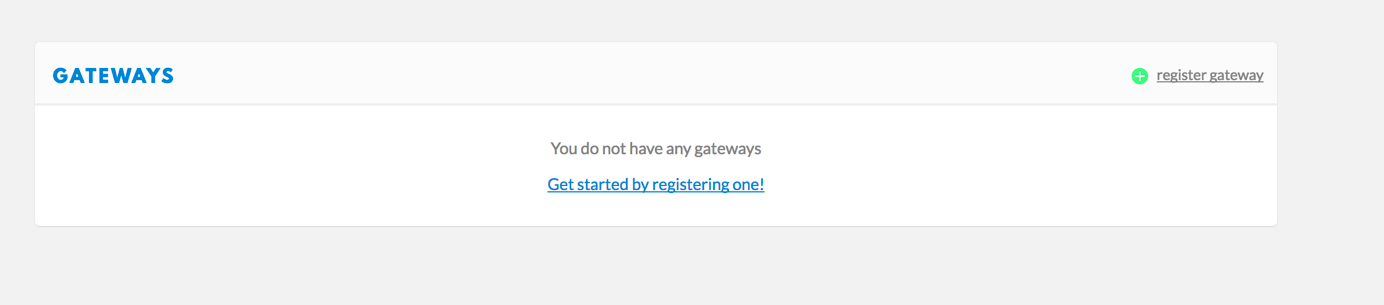
Une fois inscrit sur le site TTN, on peut accéder via l’onglet console en dessous du nom d’utilisateur, à l’interface suivante :



Cette interface va permettre de rajouter soit des applications (c’est là que l’on va pouvoir rajouter nos périphériques), soit des gateways.

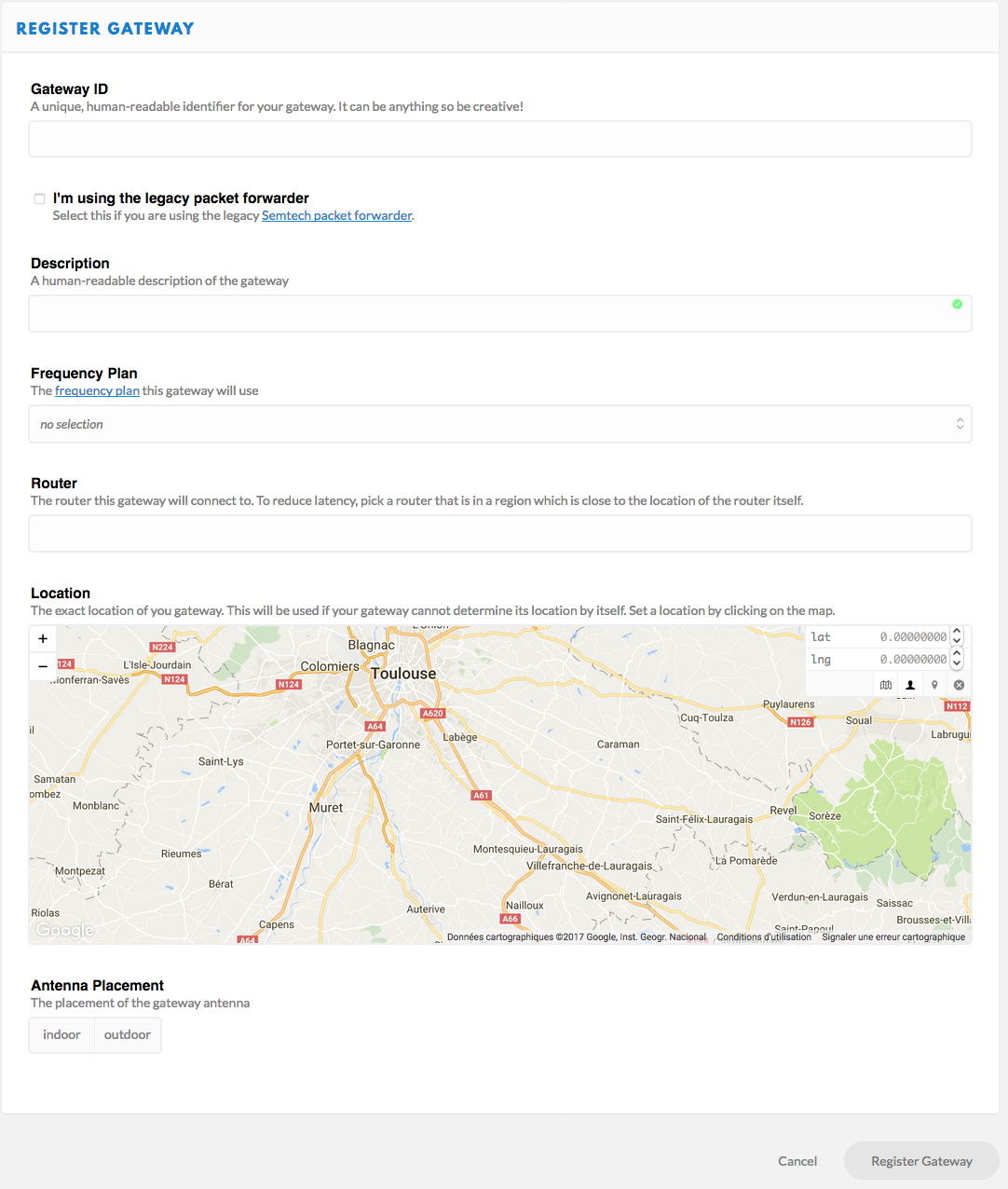
Commençons par cliquer sur « GATEWAYS »

1. **Ajouter une gateway**



En cliquant sur « GATEWAY », on arrive sur la page suivante qui va nous permettre d’enregistrer notre gateway en cliquant sur « Get started registering one »

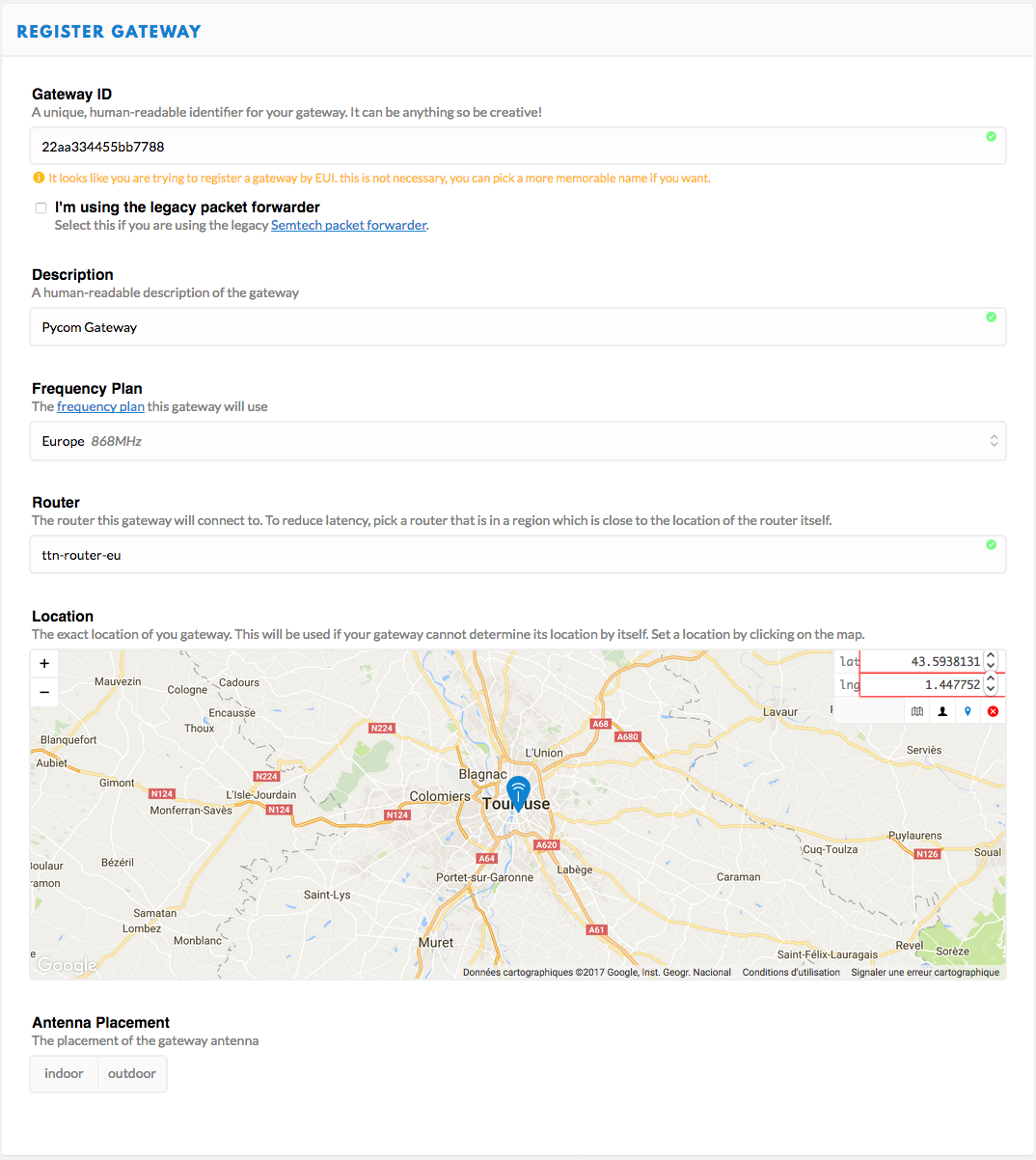
1. **Enregistrer la gateway**



Plusieurs informations nous sont demandées : l’identificateur de la gateway, une description qui sera visible des autres utilisateurs, la fréquence que nous utilisons (qui dépend de la géographie) et enfin le retour.

Ces paramètres correspondent aux paramètres de notre gateway.

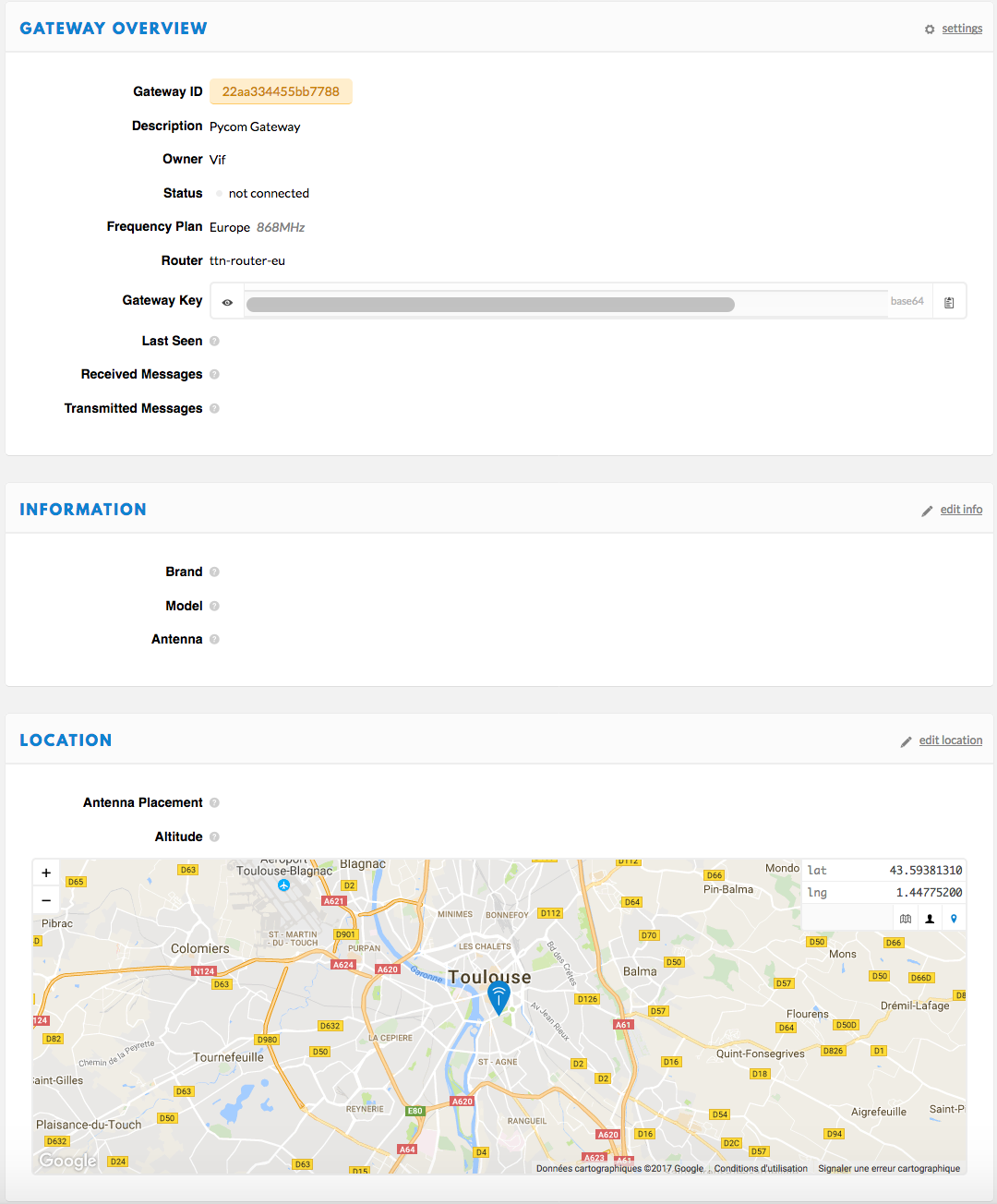
Pour finir, il nous est demandé de donner la localisation de notre gateway sur une carte, le but étant d’aider la communauté.



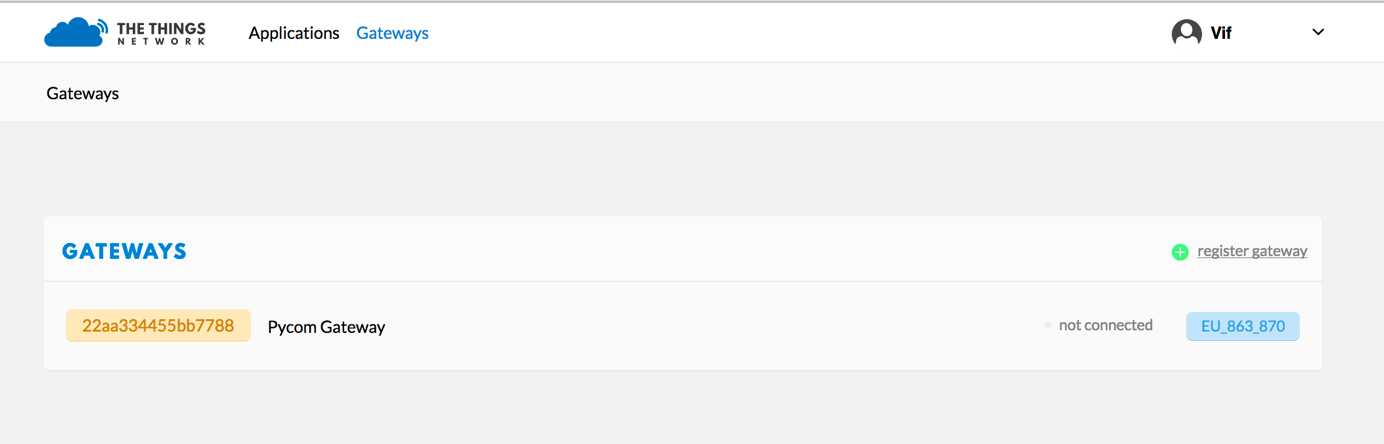
A noter que l’on n’est pas obligé de mettre l’EUID de la gateway comme identifiant.

Une fois que ces paramètres sont rentrés, nous pouvons valider et notre gateway est connectée au réseau TTN.

1. **Vue d’ensemble**



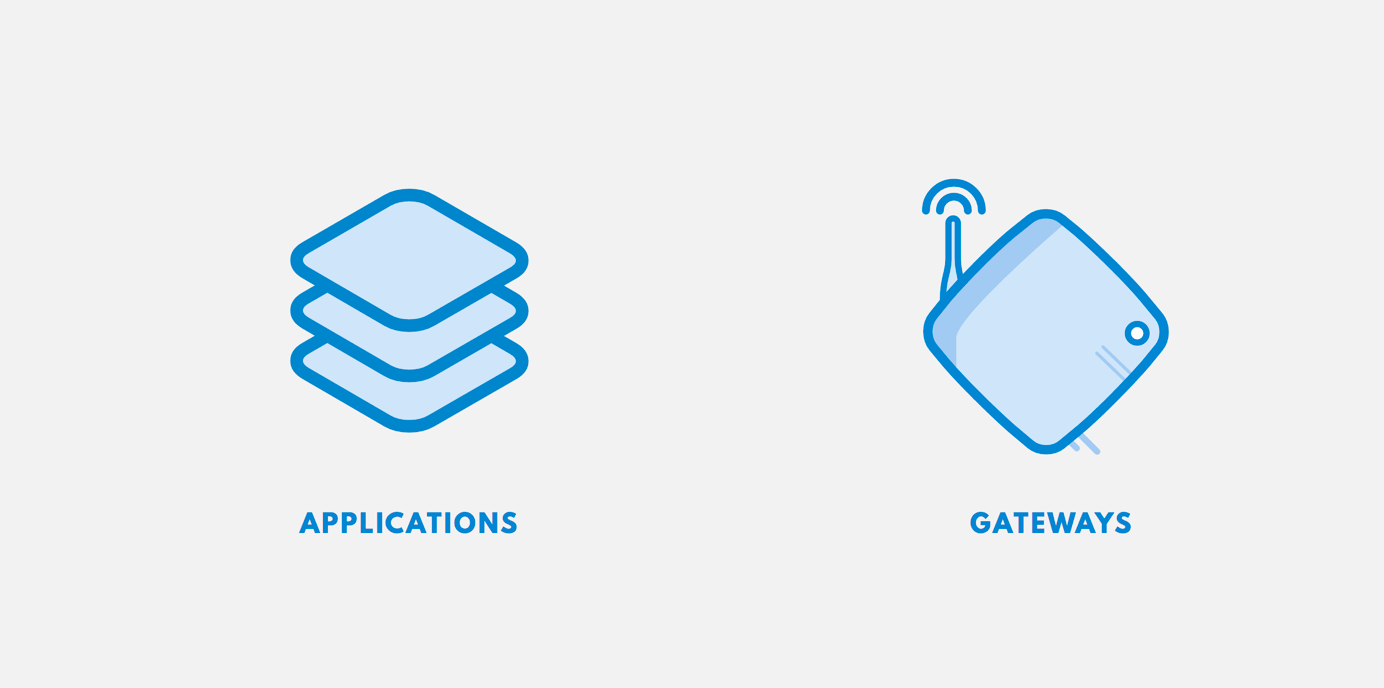
Comme nous pouvons le voir, nous avons une vue d’ensemble de notre gateway qui regroupe les éléments permettant de savoir l’état de notre gateway.

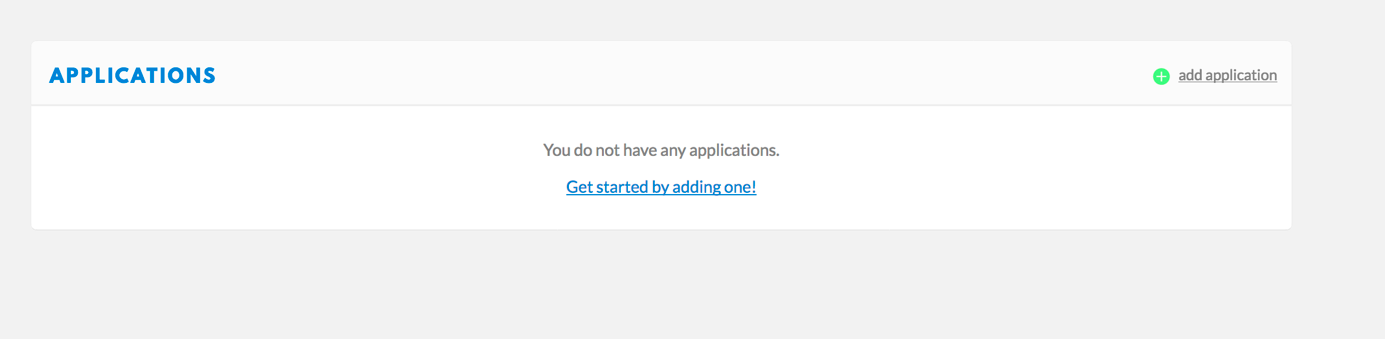


Enfin, dans l’onglet « Gateways », nous pouvons voir la nouvelle gateway qui apparaît.

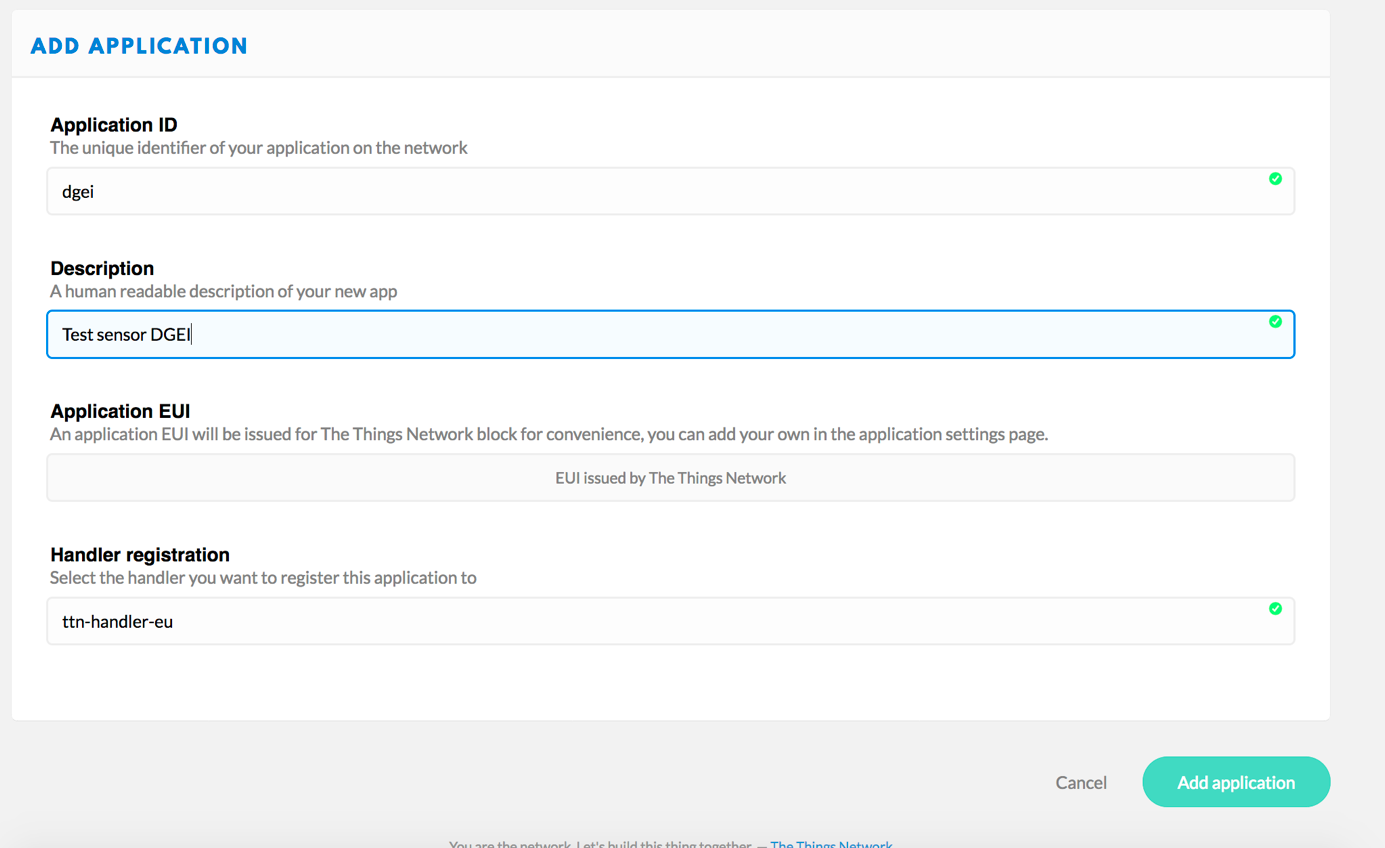
1. **Ajouter une application**

Pour connecter un capteur sur le réseau TTN, il faut commencer par créer une application.

1. **Création de l’application**

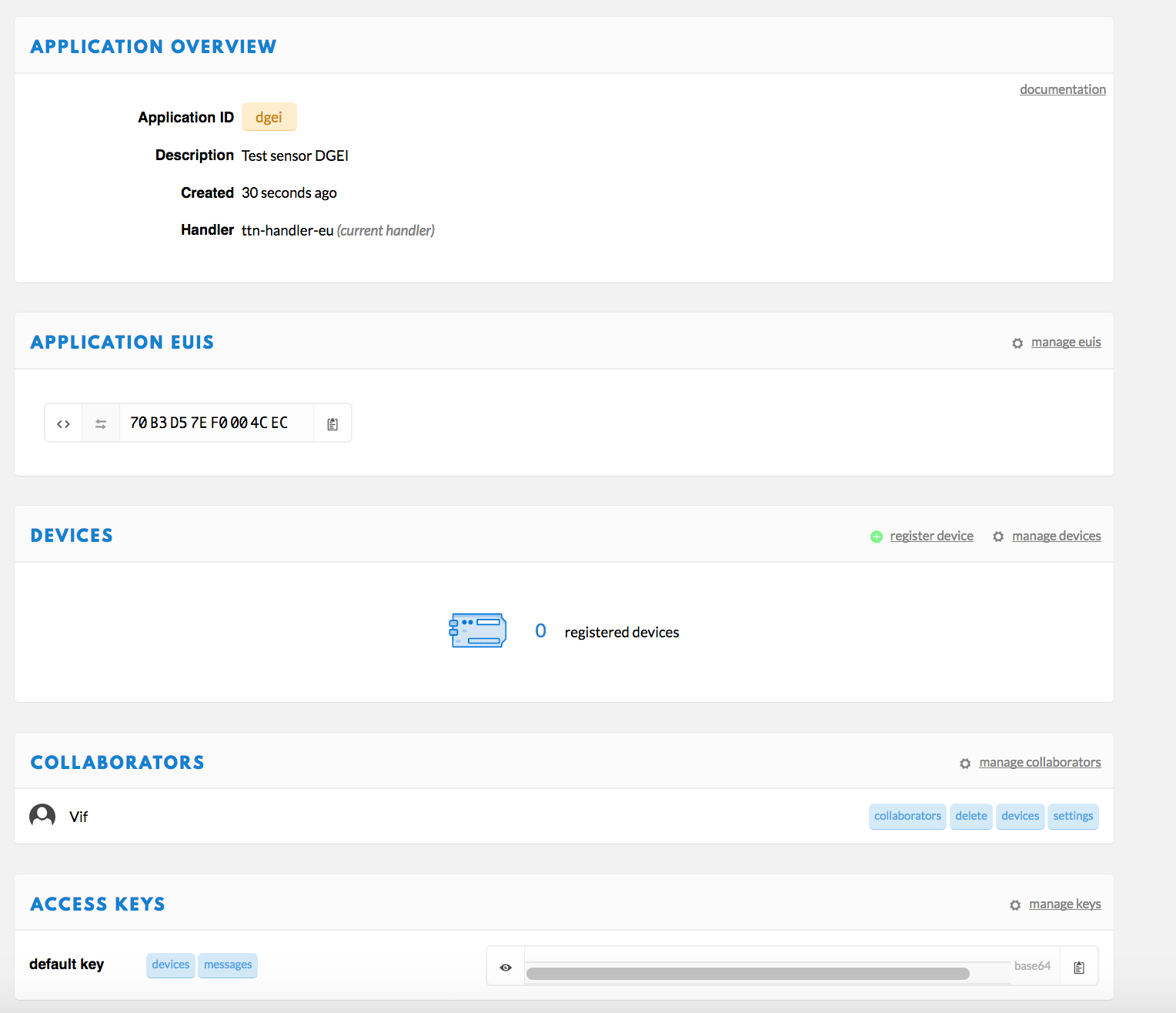


On commence par cliquer sur « APPLICATIONS » puis sur « Get started by adding one ! » si aucune application n’est créée sinon sur « add application ».



Pour ajouter une application, il faut donner un ID unique qui permettra d’identifier l’application sur le réseau, une description de l’application et enfin de « l’handler ».

L’application EUI nous est fournie par TTN, cette clef va nous permettre de connecter notre module.

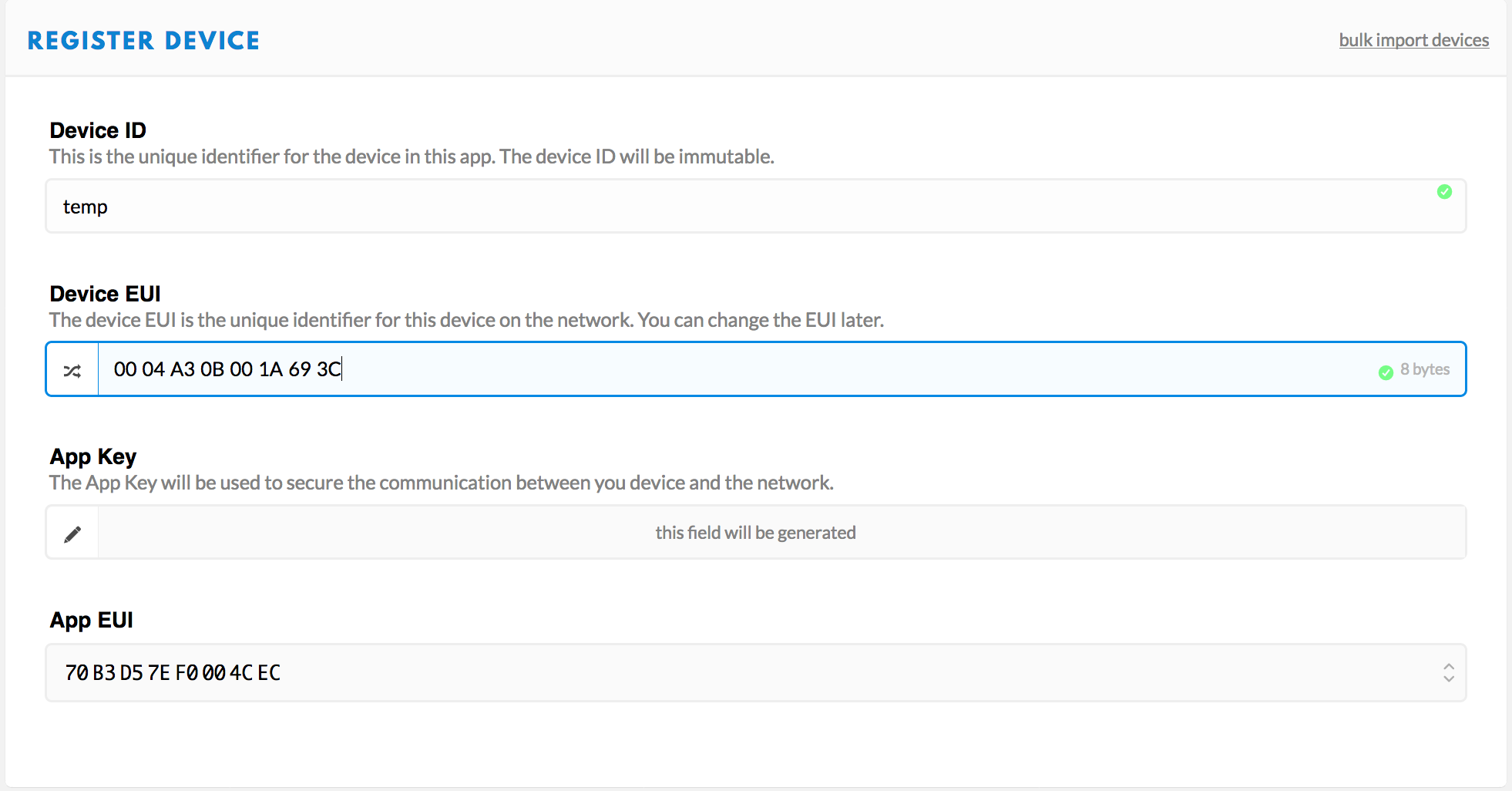


Voilà ce que l’on peut observer une fois que nous avons rempli le formulaire de l’application.

L’ACCESS KEYS permet d’accéder à notre ressource via MQTT.

1. **Enregistrement du module**

Une fois l’application enregistrée, nous pouvons enregistrer notre module pour qu’il puisse accéder à TTN. Pour ce faire il suffit de cliquer sur « register device ».

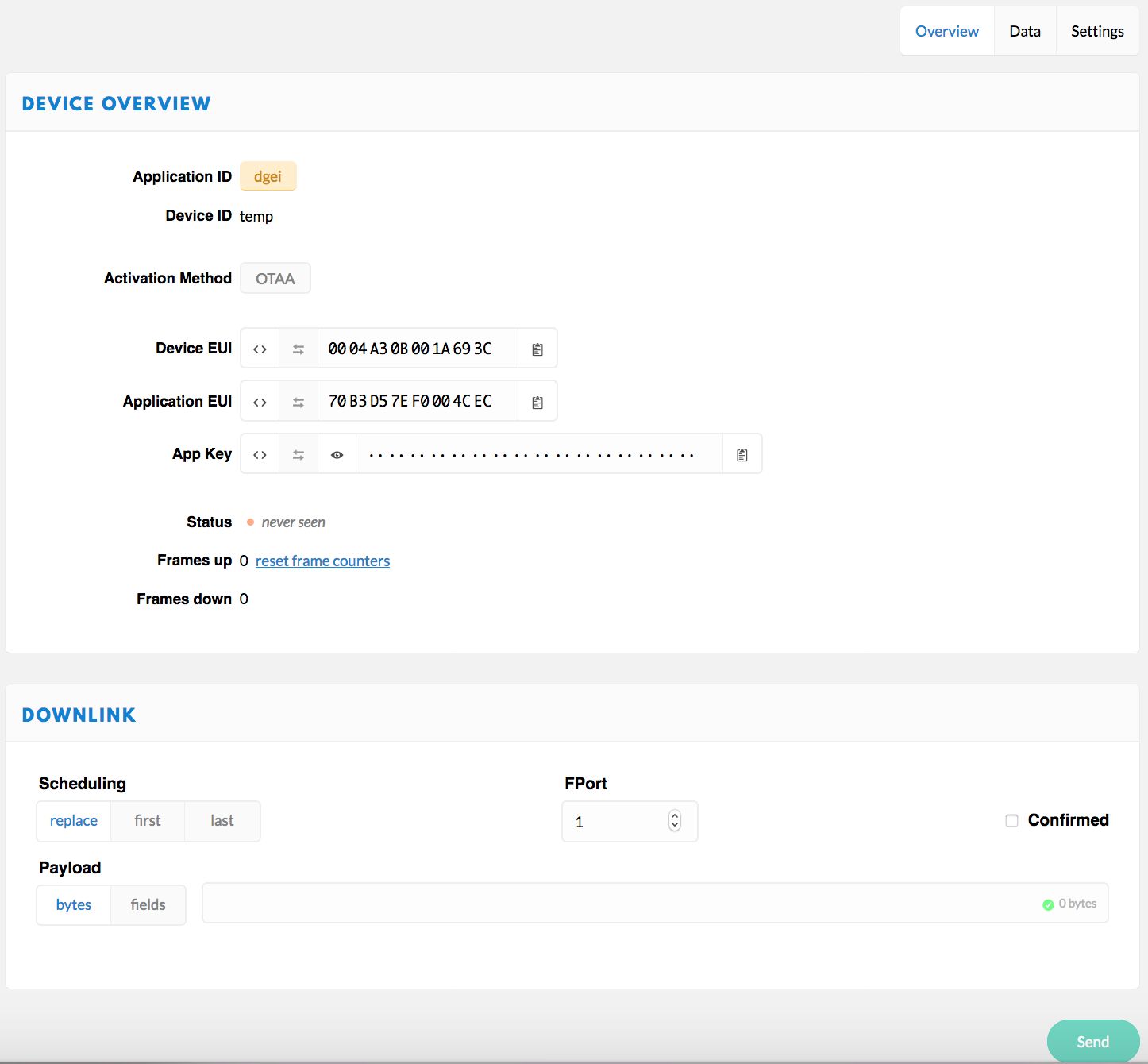


Puis de remplir les champs suivants :

* Device ID : permet d’identifier notre module au niveau de l’application
* Device EUI : permet d’identifier le module sur le réseau, clef de 8bits, donnée par le fondeur de la puce mais peut être modifiée.

« App Key » nous est donnée par TTN, cette clef va permettre à calculer un MIC permettant de joindre le réseau via la méthode d’authentification OTAA.

« L’App EUI » correspond à l’identificateur de l’application.



Une fois que nous avons validé, nous pouvons voir la vue d’ensemble sur notre module avec les clés et la méthode de connexion qui y correspond, à savoir OTAA dans ce cas-ci.

Pour se connecter, il faudra donc renseigner au module que nous devons effectuer une connexion de type OTAA avec « l’app key », « l’application EUI » et le « device EUI » correspondant aux clés données par TTN.

**ANNEX 2**

**Documentation Réseau LoRaWAN privé**

1. **La gateway**

La gateway LoRaWAN utilisée est un concentrateur iC880a USB que l’on connecte en SPI à un Raspberry Pie.

Pour pouvoir transmette les paquets de la gateway au serveur, il faut installer :

* La librairie lora\_gateway : <https://github.com/Lora-net/lora_gateway>
* L’application packet\_forwarder : <https://github.com/Lora-net/packet_forwarder>

L’application sert à transmettre les paquets en UDP au serveur, le fichier global\_conf.json permet de modifier les paramètres d’émission, de réception de la puce SX1272 mais aussi l’ID de la gateway, l’adresse du serveur et les ports.

Ce fichier est disponible à l’adresse : /packet\_forwarder/lora\_pkt\_fwd/

1. **Création d’un réseau LoRaWAN privé local**

Dans un premier temps, il faut installer :

* Mosquitto (broker MQTT): https://mosquitto.org/
* Redis server (système de gestion de base de données) : https://redis.io/
* Postgre SQL (bases de données) : <https://www.postgresql.org/>

Puis télécharger les trois parties de notre réseau LoRaWAN : lora gateway bridge, lora app server, lora server.

<https://docs.loraserver.io/lora-app-server/overview/downloads/>

<https://docs.loraserver.io/loraserver/overview/downloads/>

<https://docs.loraserver.io/lora-gateway-bridge/overview/downloads/>

Créer les bases de données :

-- create the loraserver\_ns user with password 'dbpassword'

create role loraserver\_ns with login password 'dbpassword';

-- create the loraserver\_ns database

create database loraserver\_ns with owner loraserver\_as;

-- create the loraserver\_as user

create role loraserver\_as with login password 'dbpassword';

-- create the loraserver\_as database

create database loraserver\_as with owner loraserver\_as;

Créer des certificats et clefs :

openssl req -x509 -newkey rsa:2048 -keyout key.pem -out cert.pem -days 90 –nodes

JWT : openssl rand -base64 32

Puis démarrer mosquito:

(sous macOS)  /usr/local/sbin/mosquitto -c /usr/local/etc/mosquitto/mosquitto.conf

Et enfin démarrer les trois parties de notre architecture :

./lora-gateway-bridge --udp-bind 0.0.0.0:1700

./loraserver --net-id 010203 --band EU\_863\_870 --postgres-dsn postgres://loraserver\_ns:dbpassword@localhost/loraserver\_ns?sslmode=disable --db-automigrate

./lora-app-server --postgres-dsn postgres://loraserver\_as:dbpassword@localhost/loraserver\_as?sslmode=disable --db-automigrate --http-tls-cert cert.pem --http-tls-key key.pem --jwt-secret NBtSeuEMZ1T9Rrqv6kX1xfTiISHYtf+UtyxzCWF/o+E=

Pour afficher l’aide aux commandes, faire : « ./application que l’on veut lancer –h ».

Pour plus d’informations, voir : <https://docs.loraserver.io/overview/>

1. **Création d’un réseau LoRaWAN privé**

Voir : <https://docs.loraserver.io/install/quick-install/>

Installation Redis, MQTT, PostgreSQL :

sudo apt install mosquitto mosquitto-clients redis-server redis-tools postgresql

Authentification mosquitto :

# Create a password file for your mosquitto users, starting with a “root” user.

# The “-c” parameter creates the new password file. The command will prompt for

# a new password for the user.

sudo mosquitto\_passwd -c /etc/mosquitto/pwd loraroot

# Add users for the various MQTT protocol users

sudo mosquitto\_passwd /etc/mosquitto/pwd loragw

sudo mosquitto\_passwd /etc/mosquitto/pwd loraserver

sudo mosquitto\_passwd /etc/mosquitto/pwd loraappserver

# Secure the password file

sudo chmod 600 /etc/mosquitto/pwd

Configuration Mosquitto :

allow\_anonymous false

password\_file /etc/mosquitto/pwd

cafile /etc/mosquitto/certs/ca.crt

certfile /etc/mosquitto/certs/hostname.crt

keyfile /etc/mosquitto/certs/hostname.key

sudo systemctl restart mosquitto

PostgreSQL :

sudo -u postgres psql

create role loraserver\_as with login password 'dbpassword';

create role loraserver\_ns with login password 'dbpassword';

-- create the database for the servers

create database loraserver\_as with owner loraserver\_as;

create database loraserver\_ns with owner loraserver\_ns;

-- exit psql

\q

Ajout des dépôts LoRa server :

sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 1CE2AFD36DBCCA00

sudo echo "deb https://repos.loraserver.io/ubuntu xenial testing" | sudo tee /etc/apt/sources.list.d/loraserver.list

sudo apt update

Installation du bridge :

sudo apt install lora-gateway-bridge

Modifier le fichier */etc/default/lora-gateway-bridge* pour modifier la configuration.

Installation du serveur :

sudo apt install loraserver

Modifier le fichier */etc/default/loraserver* pour modifier la configuration.

Installation de l’application :

sudo apt install lora-app-server

Modifier le fichier */etc/default/lora-app-server* pour modifier la configuration.

Lancer les services :

sudo systemctl [start/stop/reboot] [loraserver/lora-app-server/lora-gateway-bridge]

Accéder au log :

journalctl -u [loraser/lora-app-server/lora-gateway-bridge]`

1. **Réseau LoRaWAN privé via Docker**

Installer docker : <https://www.docker.com/>

1. **Installation des trois composants**

Télécharger lora-app-server, lora-gateway-bridge, loraserver sur :

<https://github.com/brocaar>

A la racine du lora-gateway-bridge, lancer :

docker-compose run --rm gatewaybridge bash

puis make build

et enfin lora-gateway-bridge

A la racine de lora-app-server, lancer :

docker-compose run --rm appserver bash

Puis :

make requirements ui-requirements

# cleanup workspace

make clean

# run the tests

make test

# compile (this will also compile the ui and generate the static files)

make build

et enfin :

lora-app-server suivit des paramètres appropriés

A la racine de loraserver, lancer :

docker-compose run --rm loraserver bash

Puis :

# run the tests

make test

# compile

make build

et enfin :

loraserver suivit des paramètres appropriés

1. **Installation 3 en 1**

Télécharger le dossier à l’adresse : <https://github.com/siscia/loraserver-docker-componse>

Puis lance *docker-compose up* à la racine de celui-ci.

1. **Réseau LoRaWAN privé via Vagrant**

Nécessite :

* Vagrant : <https://www.vagrantup.com/>
* Logiciel de virtualisation : <https://www.virtualbox.org/>

Télécharger le dossier présent à l’adresse : <https://github.com/brocaar/loraserver-setup>

Puis à la racine de ce dossier lancer : *vagrant up*

Autres commandes utiles :

# stop the vagrant machine

vagrant halt

# restart the vagrant machine

vagrant reload

# ssh into the vagrant machine

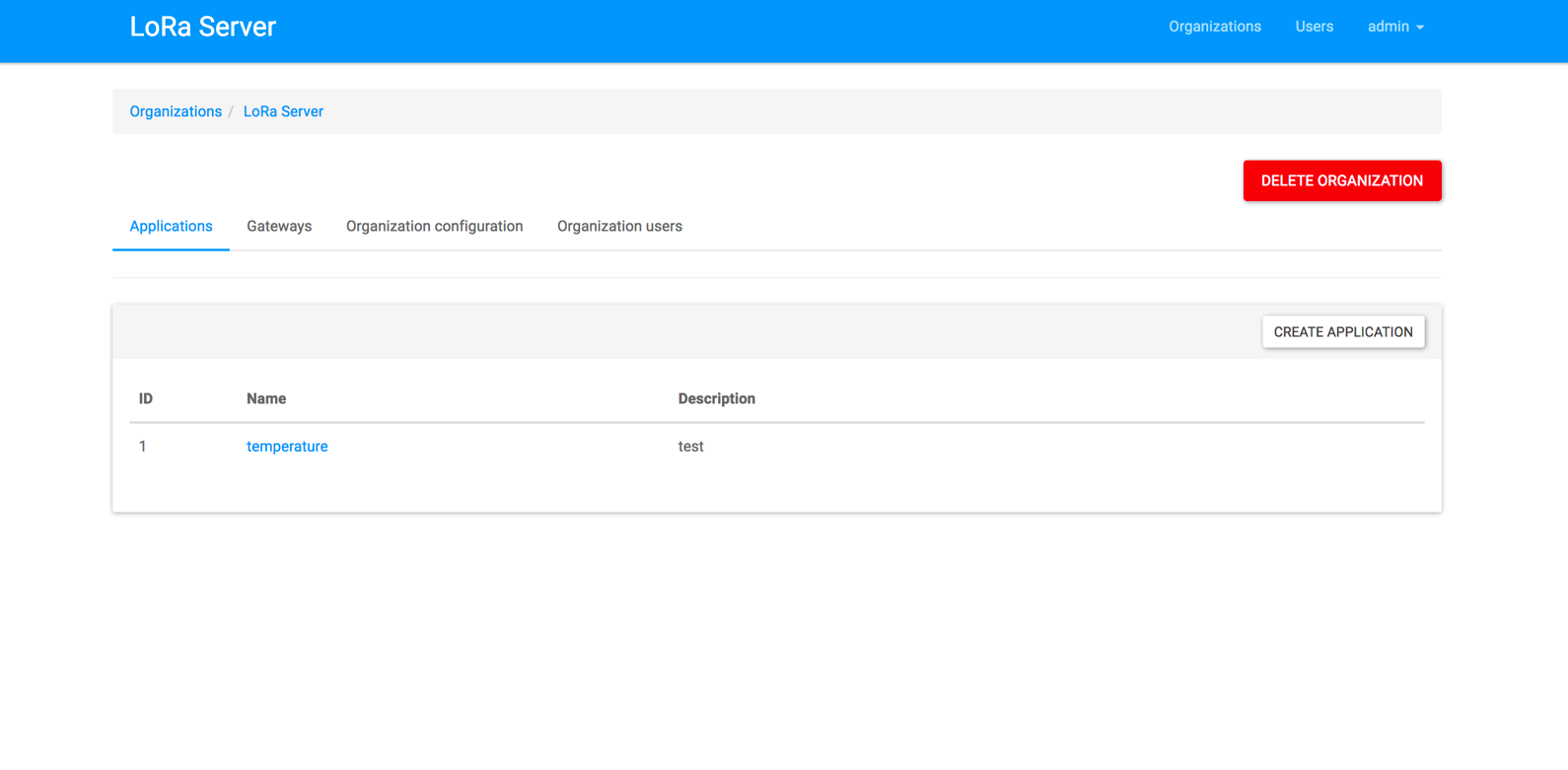
vagrant ssh

# destroy the vagrant machine

vagrant destroy

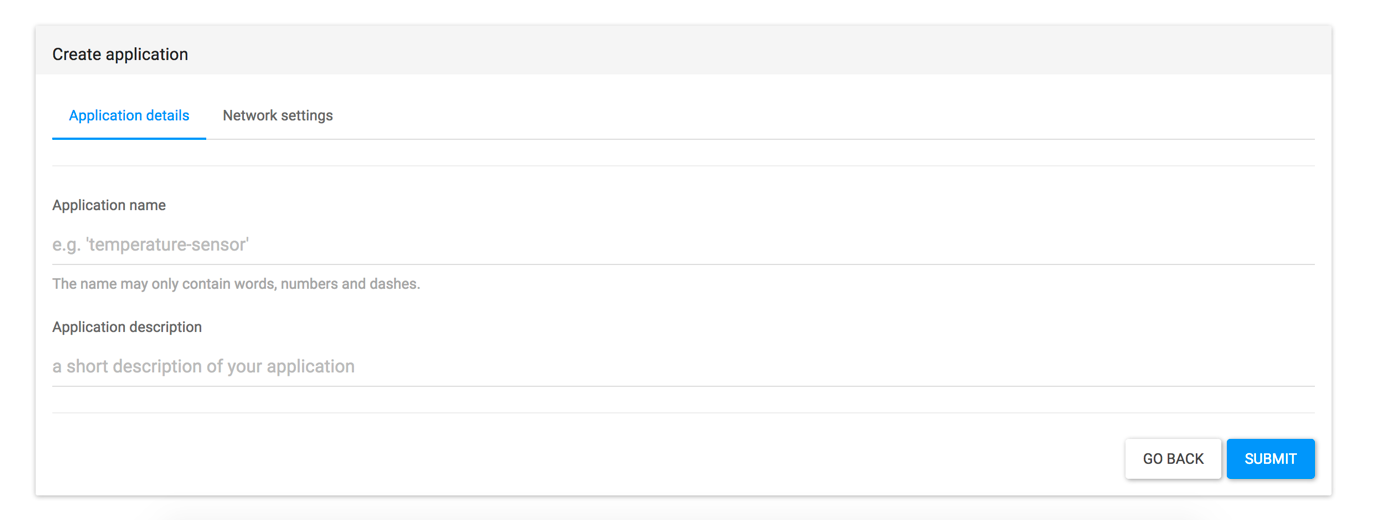
1. **Application et réception des données**

Une fois que tout est correctement paramétré à l’adresse : <https://localhost:8080>, nous avons accès à l’application web permettant d’enregistrer nos application IoT ainsi que nos modules.



Tout d’abord, il va falloir enregistrer la gateway en donnant l’ID de celle-ci à l’application.

Puis enregistrer une application :



Et enfin un nœud en donnant les trois paramètres clefs permettant l’identification du nœud sur le réseau, à savoir l’application EUI, device EUI et l’application key.

