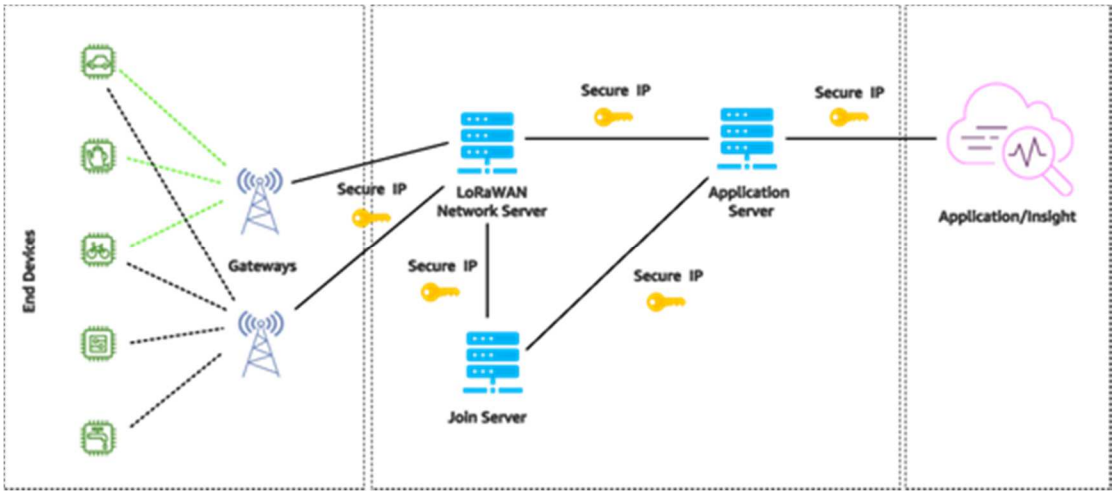
HiveMinder Deployment Instructions

Solution description and design

There are some very nice products on the market today that allow a beekeeper to closely monitor their hives. Sensors such as temperature, humidity, weight, bee counter, theft sensor, etc. Most of the products rely upon Bluetooth Low-Energy (BLE) or Wifi to communicate the data collected. The challenge with using BLE or Wifi is the range of the signal. BLE in particular is very short-ranged, especially after putting the device inside of a hive. This means a beekeeper needs to visit the hive to collect the data or the hives have be close to a wifi access point. The benefit of using BLUE, is the devices are extremely small and do not require wires. I wanted a solution that allowed for much greater wireless range and allowed data collection to happen real-time without having to physically visit each hive. LoRa/LoRaWAN is just such a system. The frequency range in the US is 915mhz and can be transmitted for miles. That range is reduced if the signal needs to pass over hills, through trees or structures but the range is substantially better then BLE/Wifi. The negative to the LoRa-based systems is that the actual sensors are 'wired' to the transmitter. Not quite as convenient as a no-wires setup like BLE/Wifi. However, I think it is worth the trade-off. Here is a graphical image of a LoRa network.

Here is a graphical image of a LoRa network.



Here is a quick summary of how data flows from the beehive all the way to your web browser. It may seem like a lot of steps but it all happens in less than a second.

- The sensor measures the temperature on a defined time interval (5 mins).
- The LoRa device transmits/broadcasts that data wirelessly on the LoRa frequency.
- A LoRa gateway receives the broadcast and passes that data along to a LoRa Network/Application server. In this case, The Things Network (TTN).
- Once the LoRa network/application server receives it, it can send it out in many different ways. Can be written to databases, sent off to other applications on the web. In this design, we'll be pulling it down locally and displaying it.

Getting Started

The following will walk you through the entire process of creating your own beehive monitoring system using LoRa sensors.

What you'll need:

- An old PC, laptop, Raspberry Pi running Ubuntu Server 20. (This guide does not provide instructions for installing Ubuntu, lots of instructions online.)
- Internet connection.
- LoRa-based sensors and a LoRa gateway. (Dragino LSN50v2 & Dragino LIG16 Indoor Gateway)
- A user account with The Things Network. (The Things Network)

Software components that will be used:

- **Grafana 8** – This will be the user interface that displays the graphs.
- **The Things Network (TTN)** – This is a web-based service (free) that acts as the LoRa network/application server. It basically receives the data from the sensor/gateway.
- **Node-Red** – This is a graphical development tool. For this purpose, it will receive data from TTN and then write it out to the Influx database.
- **InfluxDB** – This is a time-series database that will store all of the sensor data.

Cost: All of the software above is free, open-source. The only cost you will incur is for the sensors and gateway. The sensor is ~$50 which would have 2 temperature probes

High-level Steps

1. Install Grafana
2. Install InfluxDB
3. Install Node-Red
4. Configure The Things Network
5. Configure Node-Red
6. Configure LoRa gateway
7. Create Grafana Dashboards!

If you need help with something along the way, drop me an email: matthewabrandes@gmail.com

Document Revision History

Revision	Date	Description of changes
1.0	February 4th, 2022	Initial publication
1.1	February 5th, 2022	Added section at end for Configuring and Wiring the LSN50v2 unit.

Grafana Installation Instructions

(Official installation instructions: https://grafana.com/docs/grafana/latest/installation/debian/)

The following steps will walk through the process of installing the Grafana application on your server.

Add the Grafana repository

```
# sudo apt-get install -y apt-transport-https
# sudo apt-get install -y software-properties-common wget
# wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
# echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a
/etc/apt/sources.list.d/grafana.list
```

Download and install Grafana

```
# sudo apt-get update
# sudo apt-get install grafana
```

Start the Grafana applicatiom.

```
# sudo systemctl daemon-reload
# sudo systemctl start grafana-server
# sudo systemctl status grafana-server
```

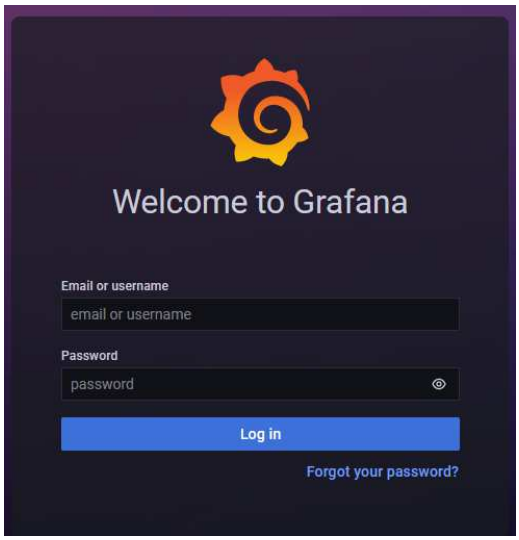
Configure the Grafana server to start at bootup.

```
# sudo systemctl enable grafana-server.service
```

Log into Grafana to verify installation and change password. (Default Login: admin/admin)

To login, open your browser and use the URL below. Grafana runs on port 3000. You will need to know what IP address the Ubuntu server has. (You can run 'ip addr' from the server CLI if you don't know.)

http://<ip address>:3000



Install InfluxDB

InfluxDB is the database that will be used. The current release is 2.1 but 1.8 is easier to work with.

Download InfluxDB 1.8

```
# curl https://dl.influxdata.com/influxdb/releases/influxdb_1.8.10_amd64.deb
--output influxdb_1.8.10_amd64.deb
```

Install and start InfluxDB

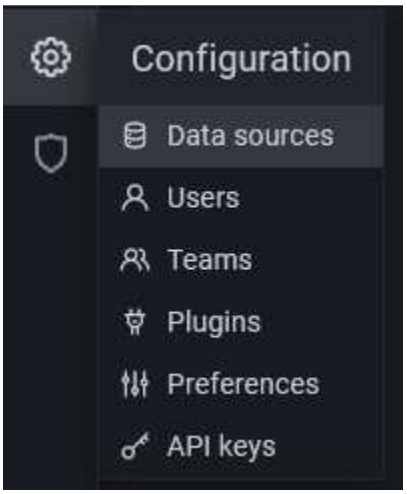
```
# sudo dpkg -i influxdb_1.8.10_amd64.deb
# sudo service influxdb start
# sudo service influxdb enable
```

Launch the Influx CLI. Create user, database and retention policy.

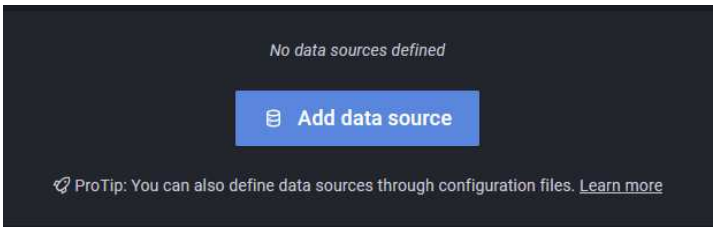
```
# influx
> CREATE USER "grafana" WITH PASSWORD 'password' WITH ALL PRIVILEGES
> CREATE DATABASE sensordata
> CREATE RETENTION POLICY sensor_rp ON sensordata DURATION INF REPLICATION 1
SHARD DURATION 52w
> exit
```

Add InfluxDB to Grafana as a Datasource

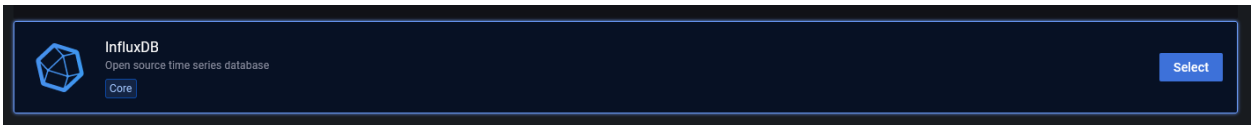
Log into Grafana, click on the Settings (Gear) icon, select Data sources.



Click the “Add data source” button.

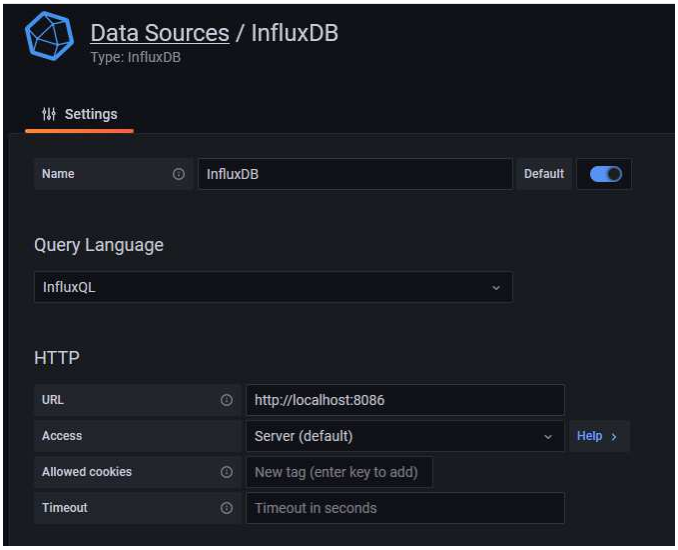


In the list of data sources, click “Select” next to InfluxDB



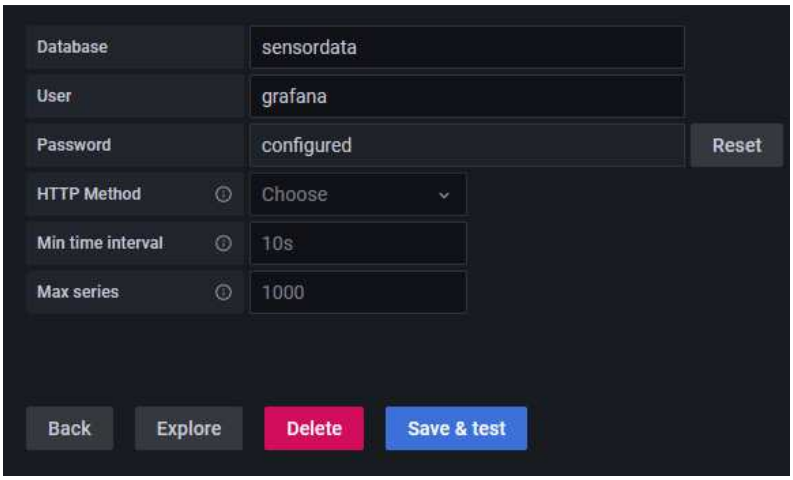
In the Settings section:

- Set the Query Language to ‘InfluxQL’.
- Set the URL to ‘http://localhost:8086’.



Scrolling down:

- Set the database name to ‘sensordata’.
- Set the User to ‘grafana’.
- Set the Password to whatever you defined above.



Click on the ‘Save & test” button. You should get a “Data source is working” confirmation message.

Install Node-Red

Install node.js

```
# curl -fsSL https://deb.nodesource.com/setup_14.x | sudo -E bash -
# sudo apt-get install -y nodejs
```

Install Node-Red

```
# sudo npm install -g --unsafe-perm node-red
```

Run node-red in background.

```
# nohup node-red &
```

*** The above command that starts Node-red, starts it in the background. If the server reboots, it will not automatically restart, you will need to run the '`nohup node-red &`' command again. (Will research way to automatically start it after reboot.)

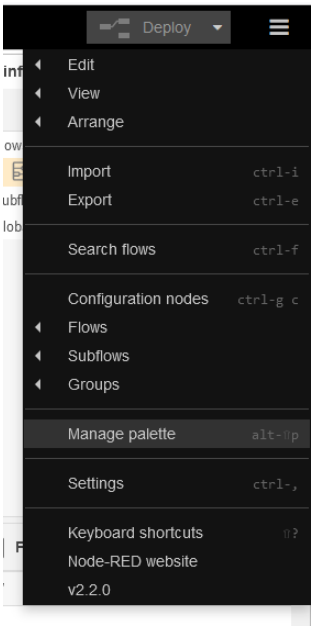
Launch Node-red interface

In your web browser, http://<IP address>:1880

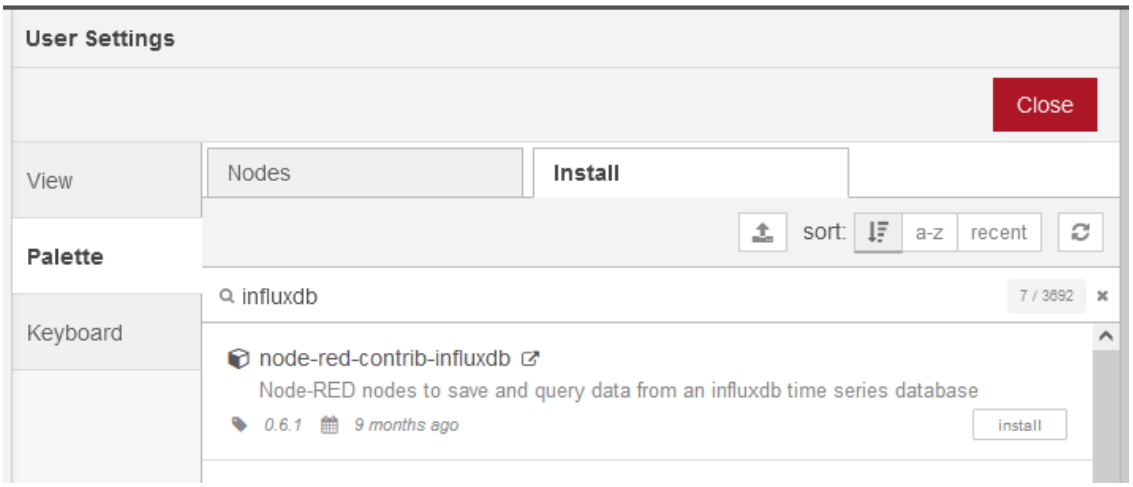
You should get this type of page. You can close out for now.

Install InfluxDB plugin

Install the InfluxDB plugin for Node-red. Open up the menu from the upper-right corner and select "Manage palette".



Click the Install tab, search for Influxdb and click Install. After installation, click Close.



Configure The Things Network (TTN)

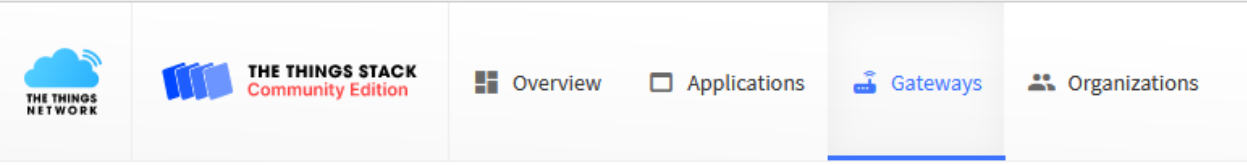
We need to setup TTN to receive data from the sensors and gateway. Assumption is that you have already created an account.

Add your LoRa gateway to TTN

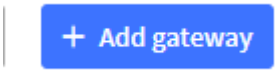
You will need to locate the EUI of your gateway. This could be on a sticker or paperwork that came with your gateway. If not, you might need to log into the gateway management interface to obtain it. (Configuring the gateway is covered a bit later if you need to skip ahead.)

Log into the TTN Console: https://nam1.cloud.thethings.network/console/

Select “Gateways” from the menu across the top.

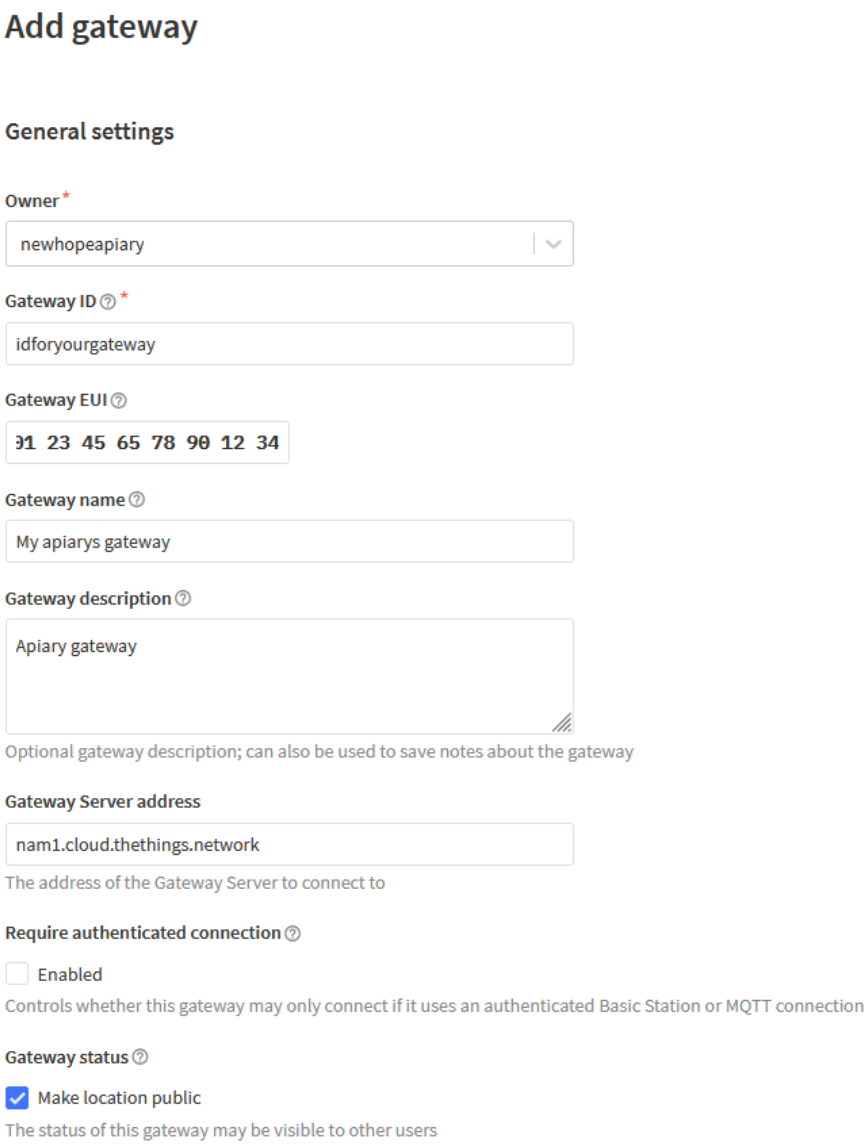


Click the “+ Add gateway” button.



On the “Add gateway” screen.

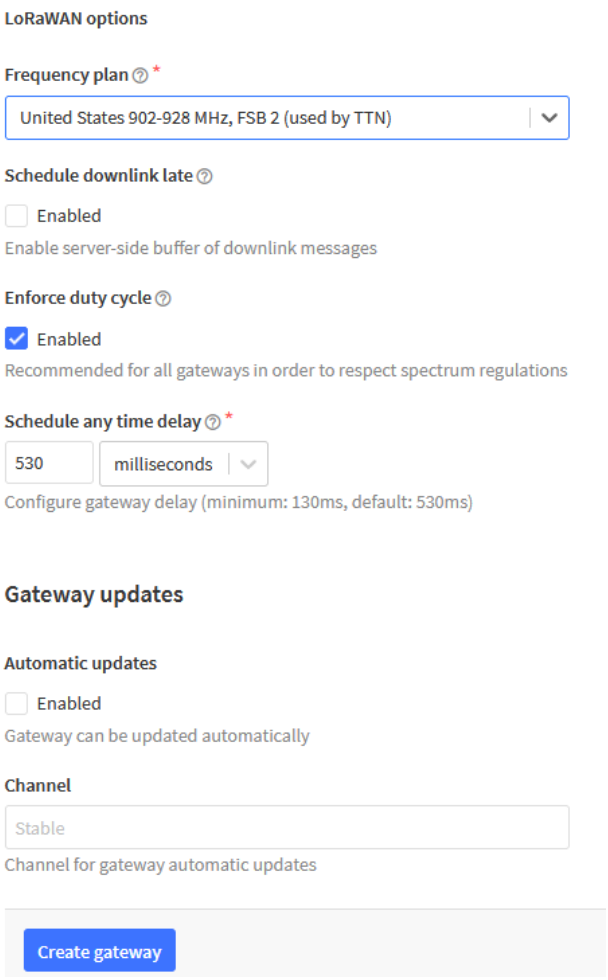
- Enter a Gateway ID. This will be a globally unique ID. Text only, no spaces allowed.
- Enter the Gateway EUI from packaging or management interface.
- Enter a Gateway name. This is just a friendly name for your purposes.
- Enter a Description if you like, not required.



Scrolling down…

Set the Frequency plan to “US 902-928 MHz, FSB2”

Click “Create Gateway”.



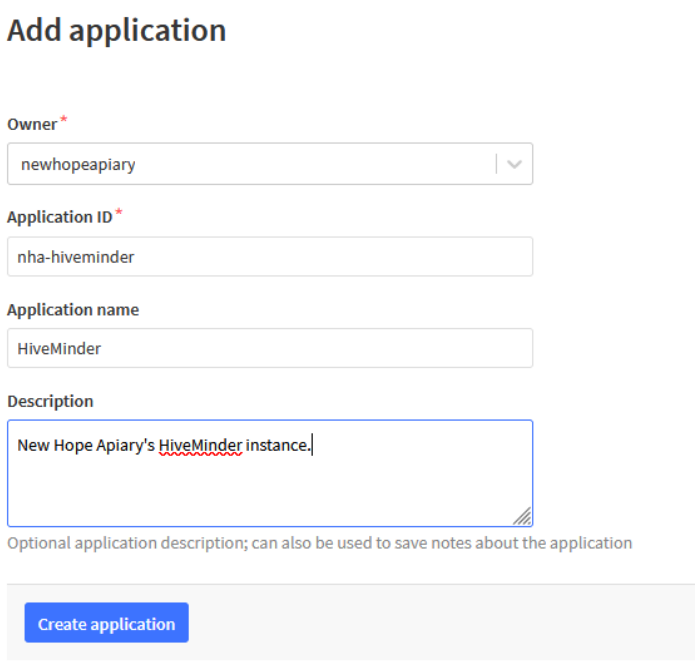
Create an application

An application is merely a collection of the devices that will be used and the integrations for them.

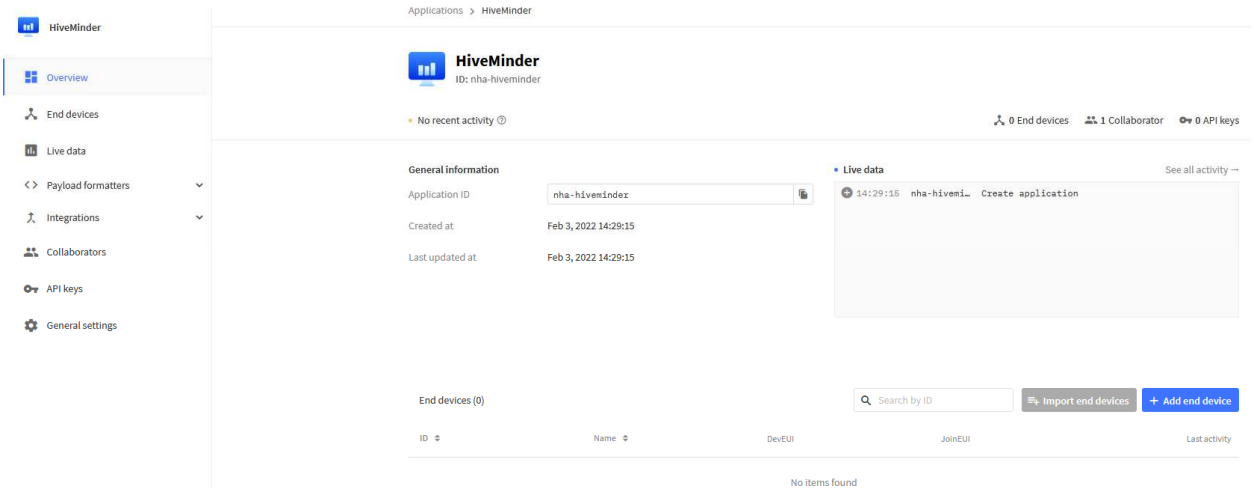
Enter an Application ID, this will be a globally unique identifier.

Enter an Application Name & Description.

Click “Create Application”.



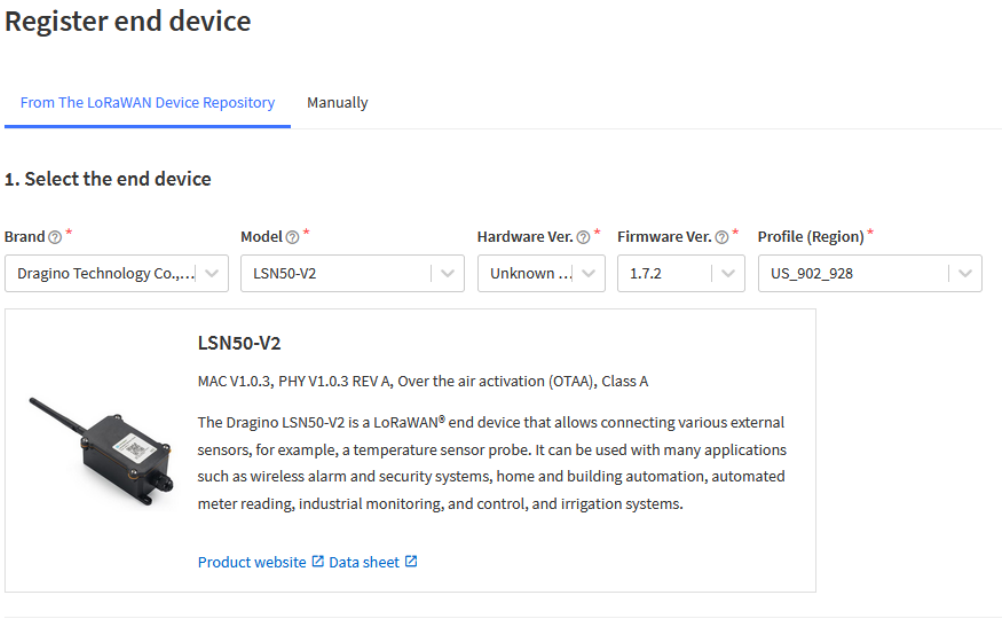
This is what the Application screen will look like. New sensor devices will be added here.



Register the LoRa sensor device(s)

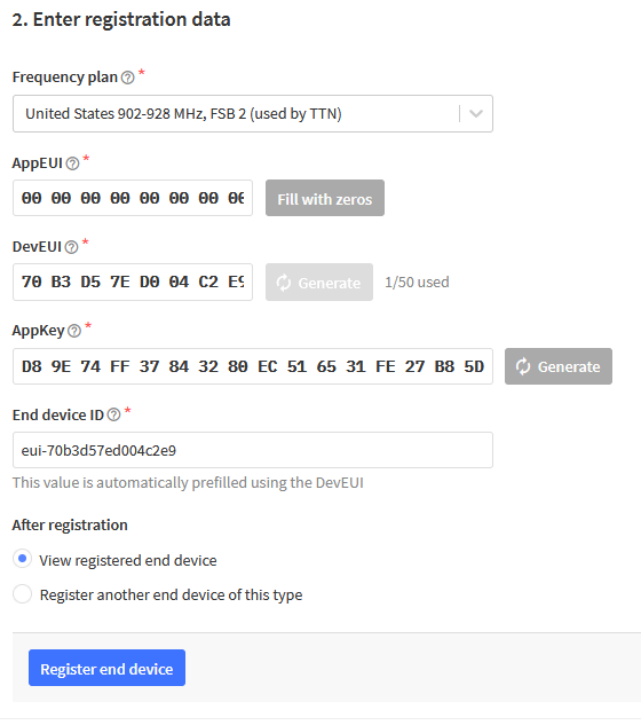
Each LoRa sensor you have will need to be configured in TTN. You will need a series of values from the device manufacturer, this should be printed on a label on the box or device. (For the LSN50v2, it is on a lable on the box it came in, take a picture, don't lose the box!)

From the TTN Application page, click on "+ Add end device". TTN knows about most certified LoRa products, so you can select your Brand, Model, etc.



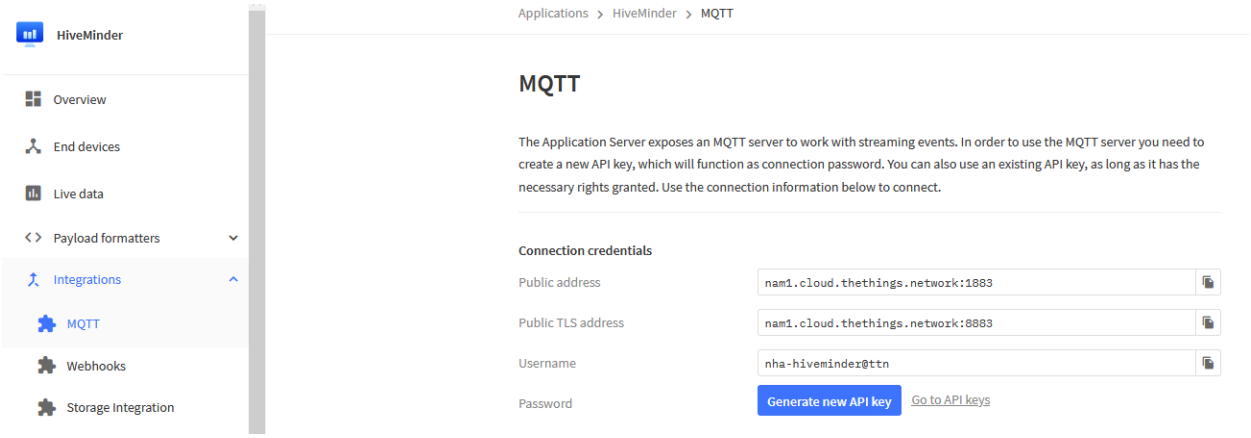
Select the US 902-928, FSB2 Frequency plan.

Enter the AppEUI, DevEUI and AppKey from the manufacturer and then click “Register end device”.



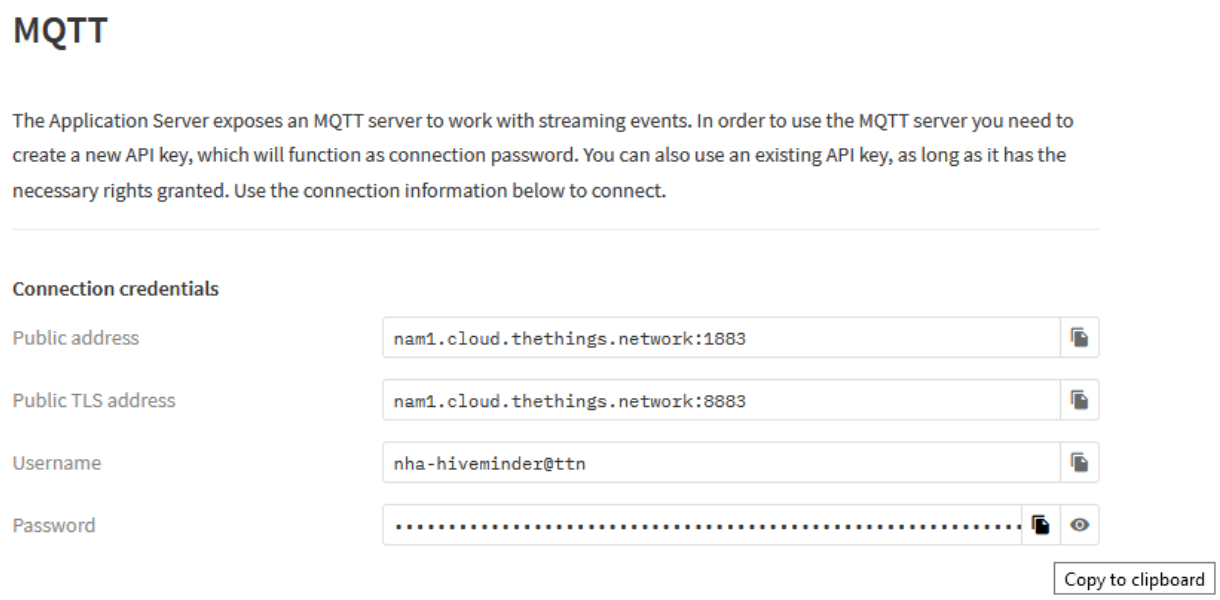
Configure MQTT integration

Using the menu on the left, expand the “Integrations” option and select “MQTT”. You can make note of the Public address, it will be used in the Node-Red configuration.



Click on the “Generate new API key”

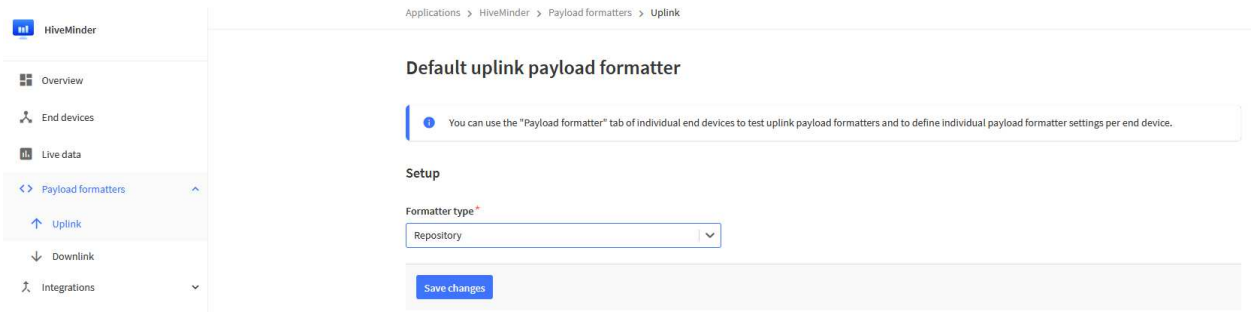
Make sure you copy the password and paste it in notepad for use in a later step. (Use the “copy to clipboard’ icon next to the password”)



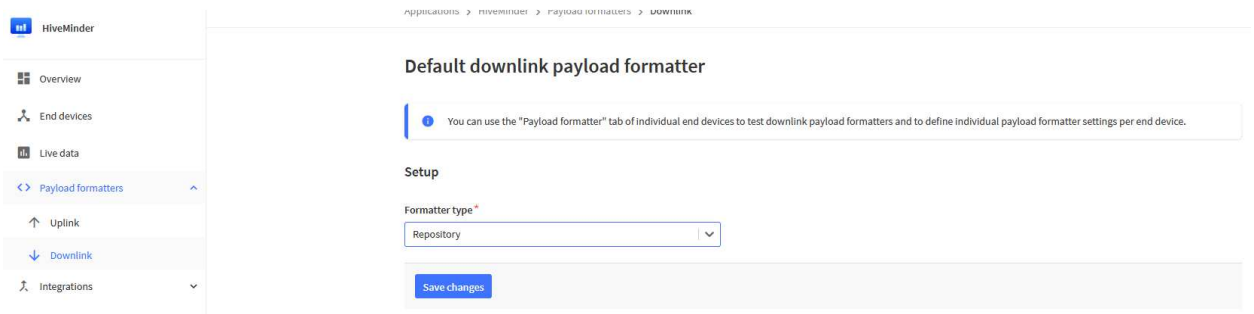
Configure Payload Formatters

Every LoRa sensor has different 'payloads' of data, so you need to tell TTN how to read the payload. For known devices (like Dragino) it already has a repository of payload definitions, so you just need to tell it to use that.

Expand the "Payload formatters" menu item, select Uplink. Select "Repository" from the pull-down and click Save Changes.



Select the Downlink menu item. Select "Repository" from the pull-down and click Save Changes.



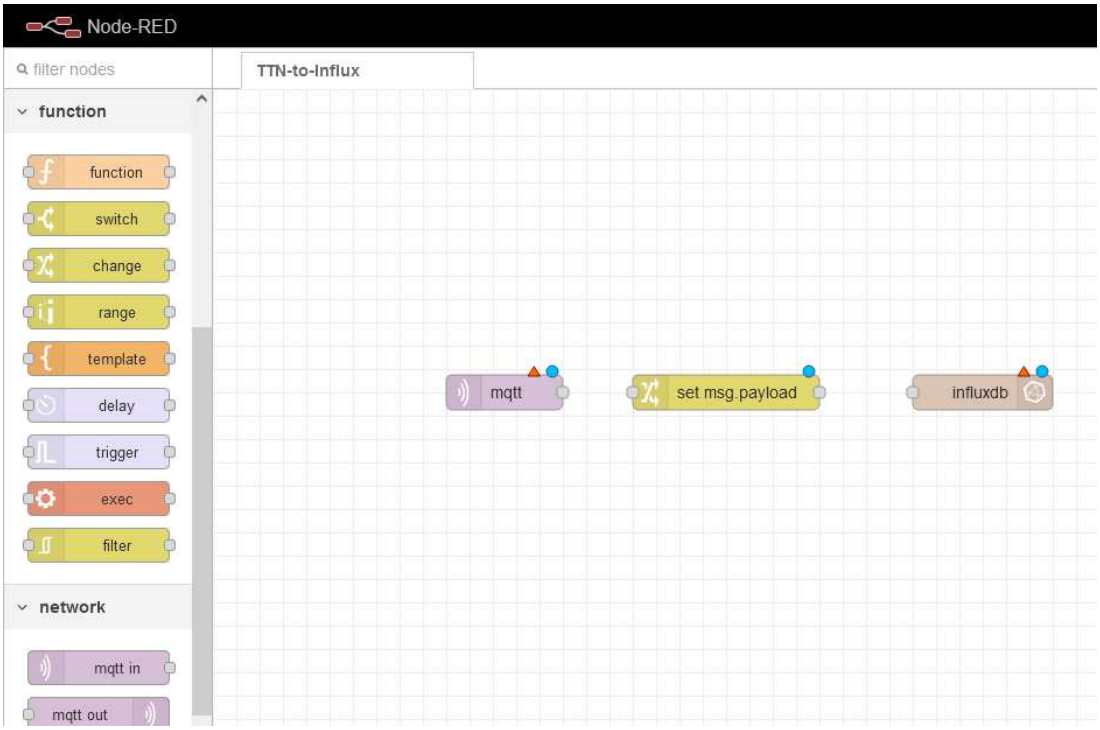
That completes the TTN configuration! Once the sensors are gateway are powered up, we can come back and verify TTN is seeing them.

Configure Node-Red

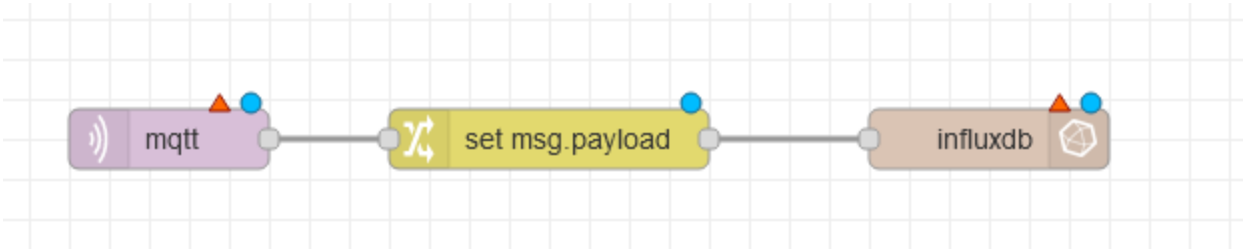
Node-Red is the application that will receive the sensor data from TTN and then place it into a database. Once in the database, you view that data with Grafana. Node-Red is a graphical design tool, so you will use your mouse to click-n-drag items and to connect them together with lines.

Open Node-Red in your web browser: http://<ip address>:1880

From the left menu, click-hold-drag an '*mqtt in'* node, a '*change*' node and an '*influxdb out*' node onto the palette.



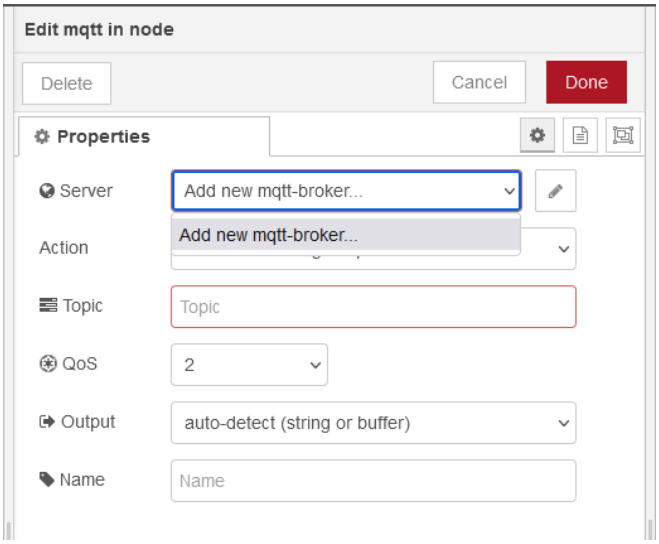
Connect the nodes together. (Use mouse cursor to draw line from the points on the nodes)



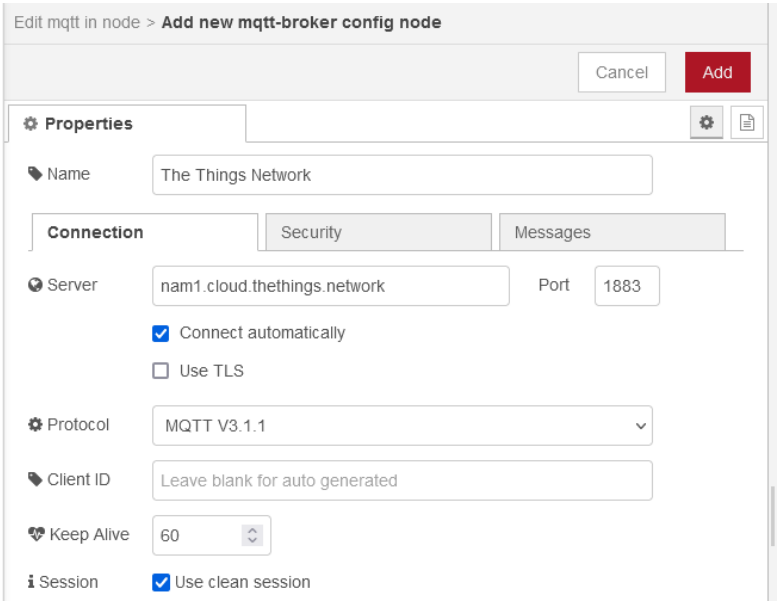
Configure MQTT

Double-click on the 

node to open the properties. Click the Pencil icon next to the Server box to add a new mqtt-broker (TTN).

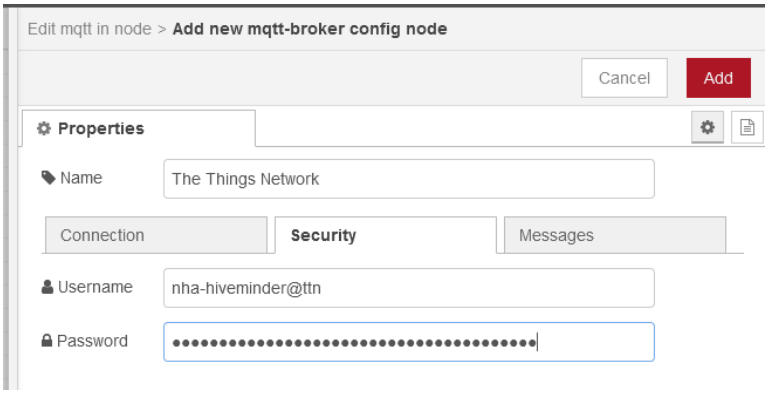


- Enter a name for the server (The Things Network)
- Enter 'name1.cloud.thethings.network' for the Server.



Click the Security Tab

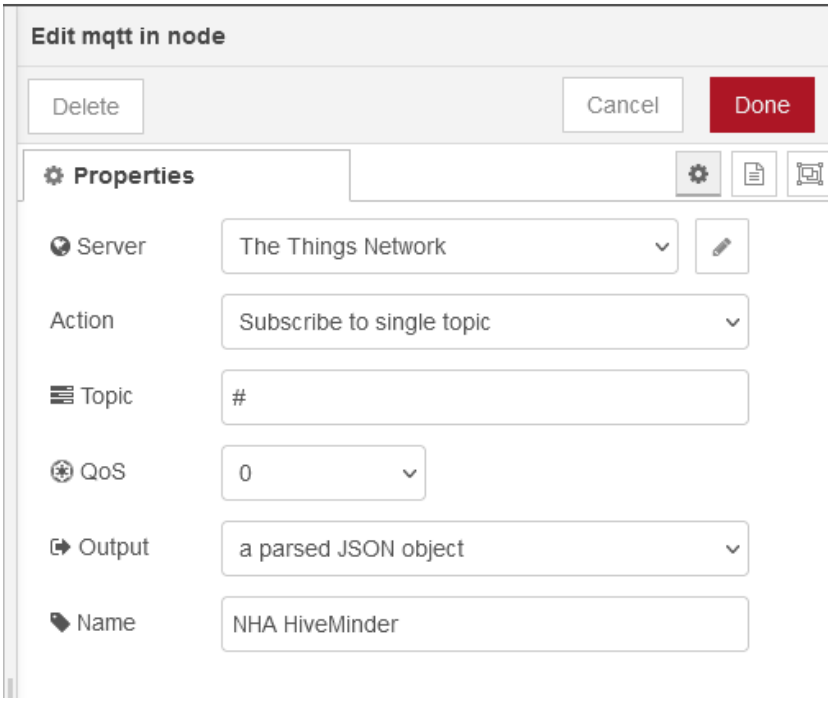
Enter your username and API token then Add. This is found on your TTN dashboard. Applications, Integrations, MQTT. (You should have copied the token to Notepad… copy/paste here.)



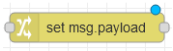
Back on the “Connection” tab.

- Enter “#” for the Topic.
- Set QoS to “0”
- Set Output to “A parsed JSON object”
- Assign a name for this node.

Click Done.

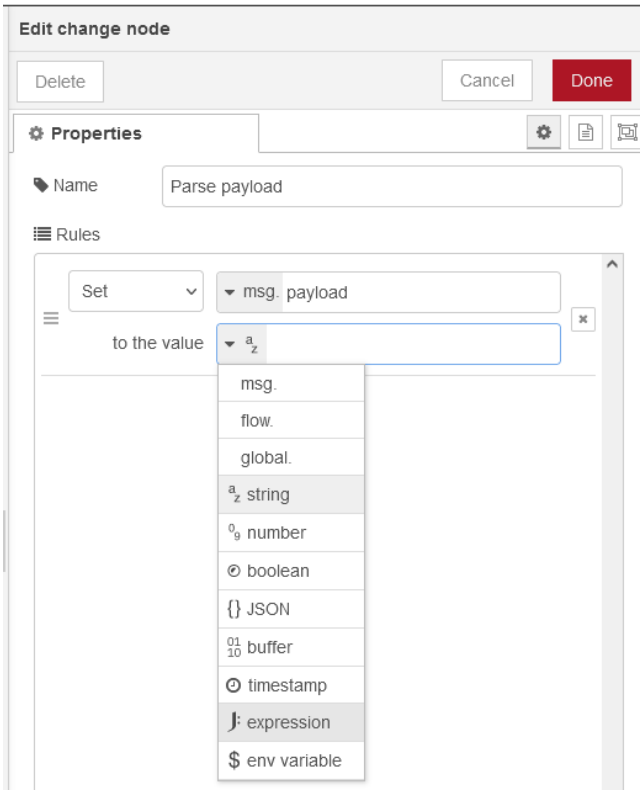


Configure the Change node.

Double click on the Change Node. 

Enter ‘Parse payload’ for the Name.

Pull down the selection arrow next to “to the value” and select “J: expression”.



Click on the  and paste in the following:

```
[
    {
        "TempUpper": payload.uplink_message.decoded_payload.TempC1,
        "TempLower": payload.uplink_message.decoded_payload.TempC2
    },{
        "hive": payload.end_device_ids.device_id
    }
]
```

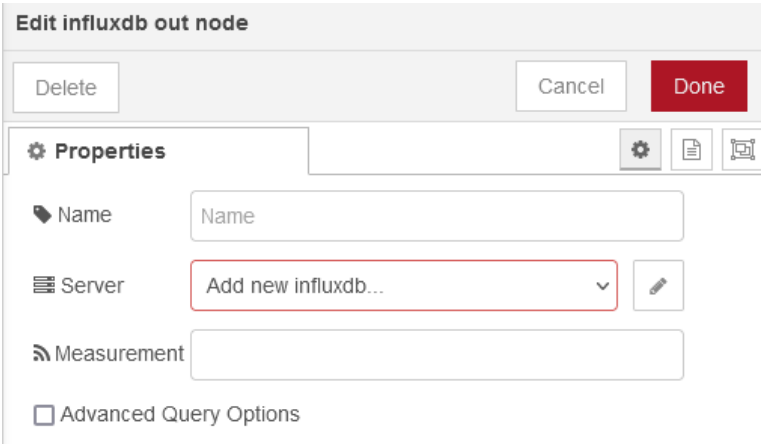
It should look like this:

Click DONE and DONE again.

Configure the InfluxDB node.

Double click the InfluxDB Out node.

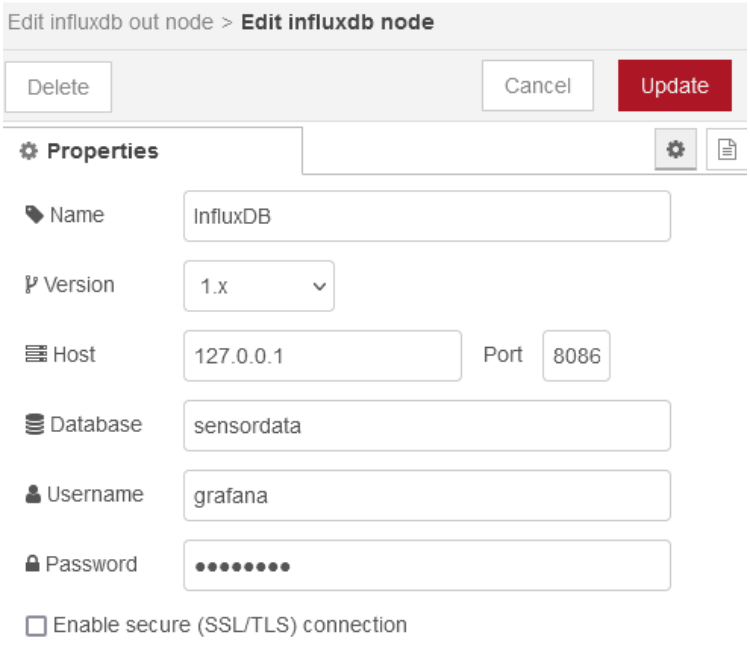
Click the pencil icon next to server to 'Add a new influxdb'.



- Enter 'InfluxDB' for the name.

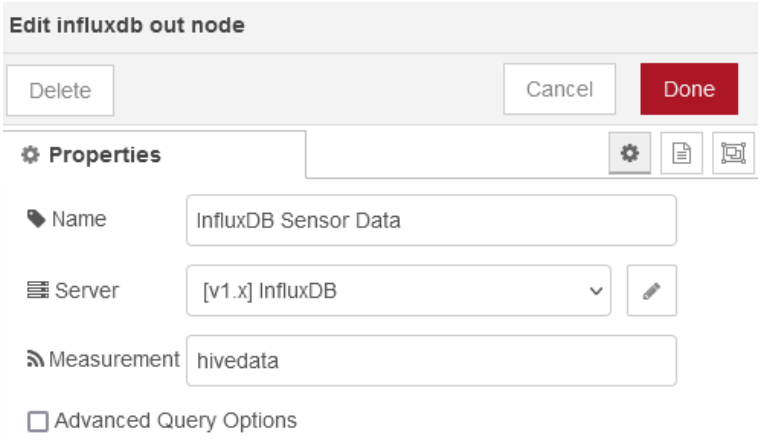
- Select version ‘1.x’.
- The host should be 127.0.0.1 Port 8086
- Enter ‘sensordata’ for the Database
- Enter ‘grafana’ for the username
- Enter the password for the grafana user. (Created previously)

Click Update.



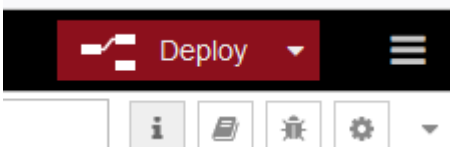
Create new server.

- Define a name.
- Select version 1.x
- Enter ‘hivedata’ for the Measurement. (This is the table data will go to.
- Click Done.

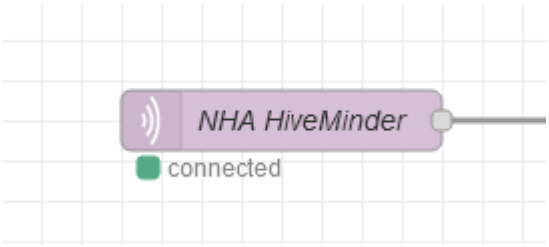


Deploy the configuration

Click Deploy in the upper right corner.



Make sure you see that the MQTT node says ‘connected’. This means it is talking to TTN successfully.



Configure Sensor(s) and Gateway

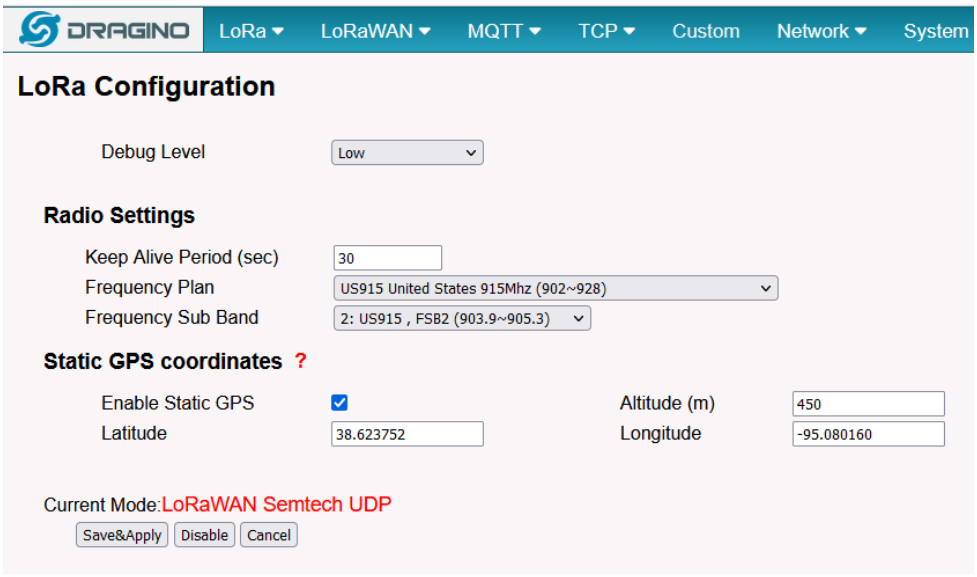
If you haven't already, now we can configure the LoRa gateway. You will need to refer to your gateway instructions on how to log into the management interface and configure in. In the screen shots below, this is from a Dragino DLOS8 Outdoor antenna. I used the Wifi connection it offers to attach to it initially.

Under the LoRa menu, you need setup just a few things:

- Frequency Plan to 'US915'
- Frequency Sub Band to '2'
- If the gateway doesn't have GPS, you can set the GPS coordinates and Altitude.

Click Save & Apply.

LoRa Settings:



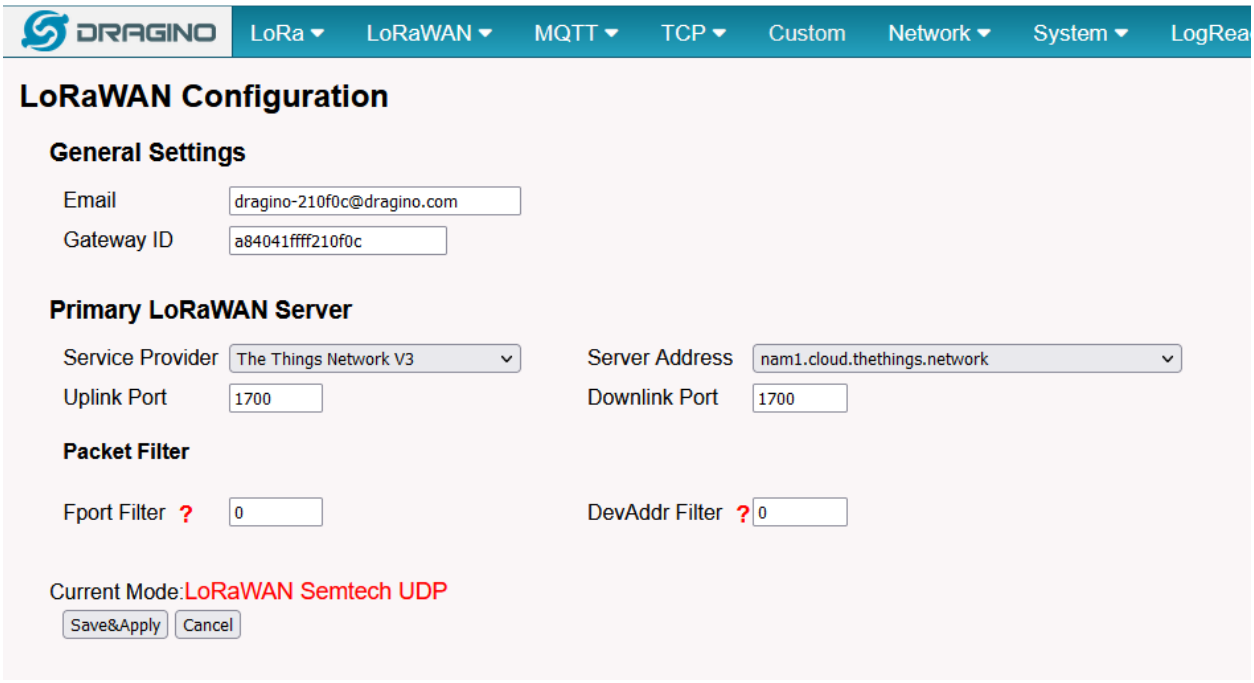
Under the LoRaWAN settings is where you will configure the TTN network.

- Service Provider set to “The Things Network v3”
- Server Address set to “nam1.cloud.thethings.network”

Click Save & Apply.

(Note: If you need the Gateway EUI for TTN, that is the value in the Gateway ID field.)

LoRaWAN Settings:



Create Grafana Dashbaord

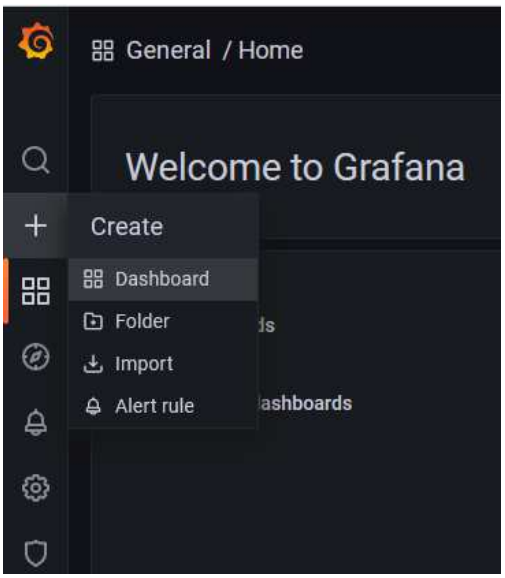
At this point we should have data flowing into the database and it is time to display it!

Log into the Grafana application.

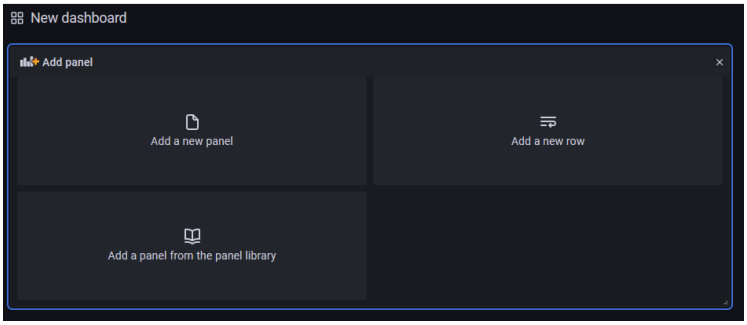
Create a Dashboard

Dashboards are a collection of panel's. Basically, graphs.

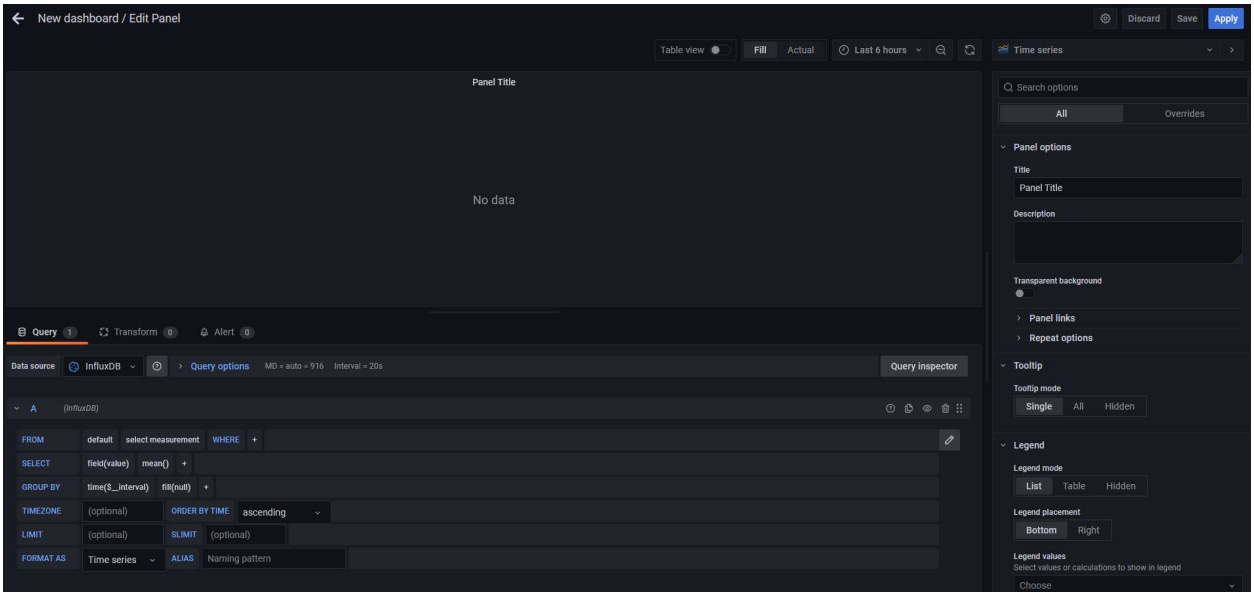
Click the '+' sign and select 'Dashboard'.



Click on "Add a new panel"



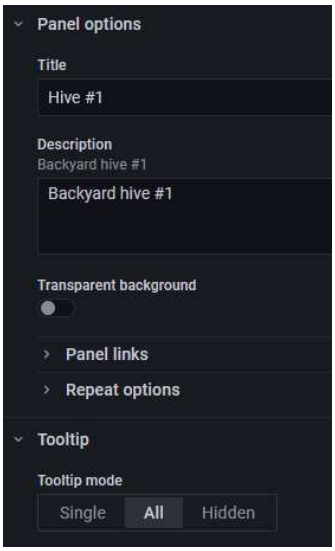
The new panel screen looks complicated… we'll break it down into pieces.



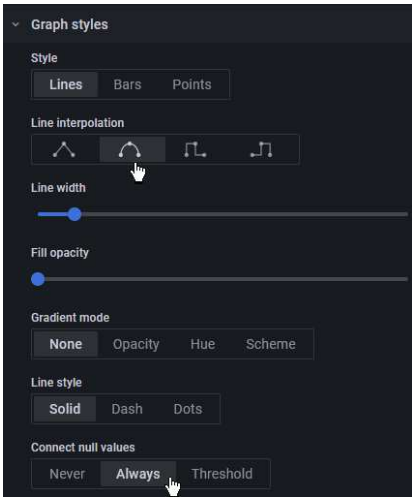
Panel configuration

Starting along the right side of the screen with Panel Options.

- Give the panel a name, this will be the hive name.
- Select "All" on the Tooltip mode, this will make it show the actual values when you mouse over the graph.

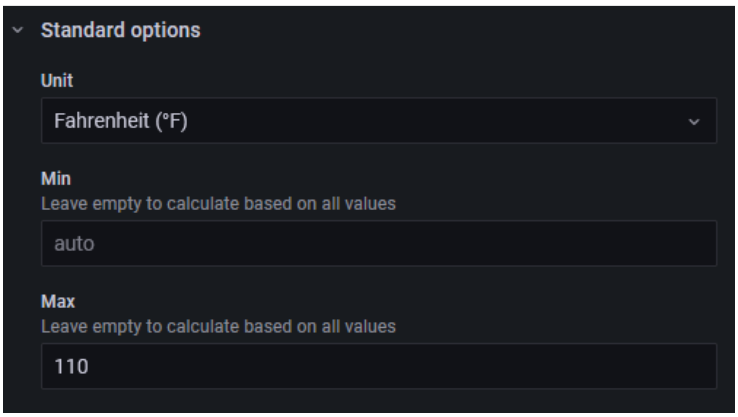


- Under the "Graph Styles" section, I like to select the Smooth option.
- Under the "Connect NULL values" select "Always".



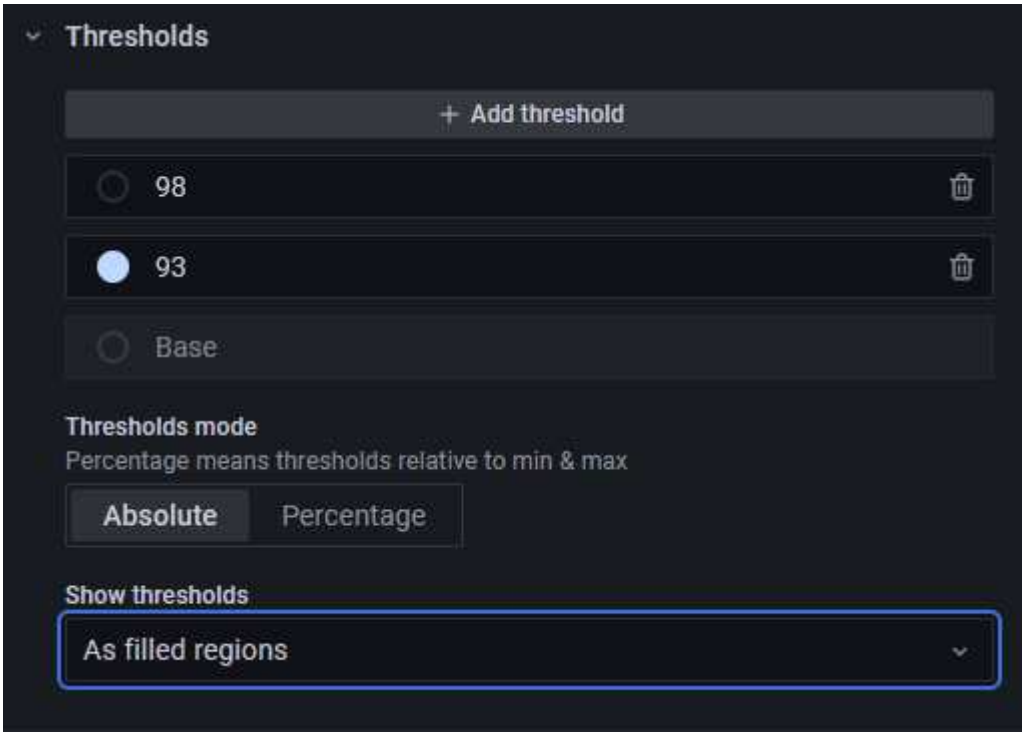
Under “Standard options”, select Fahrenheit as the Unit.

I also like to set the Max temperature value to 110. (I don’t set the minimum because you can also graph the outside temperature on the same graph and it can get COLD out.)



Under “Thresholds” – This will create a “Brood Temperature Zone” band on the graph.

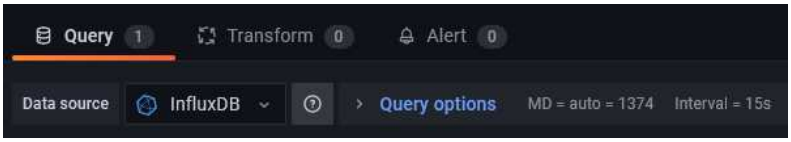
- Change the Base to be Transparent color.
- Create/modify a ‘93’ threshold and pick the color of your choosing.
- Create a ‘98’ threshold and set it to Transparent.
- Select “As filled regions” under Show thresholds.



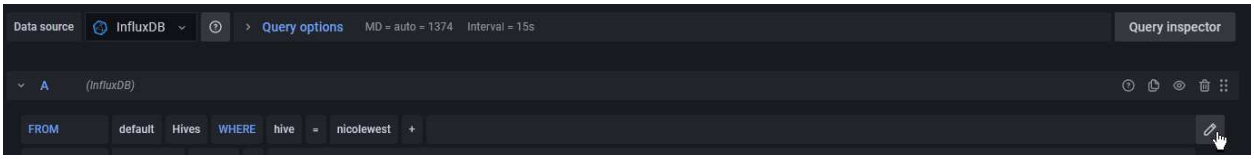
Build a query.

Now that we have the panel configuration done, we need to feed it some data. In the lower-left part of the screen is where we develop our search/query parameters for the database.

Make sure the Data source is “InfluxDB”



We need to switch from the query builder to the raw query setup. Click on the pencil icon on the right side of the query. (See hand in picture)

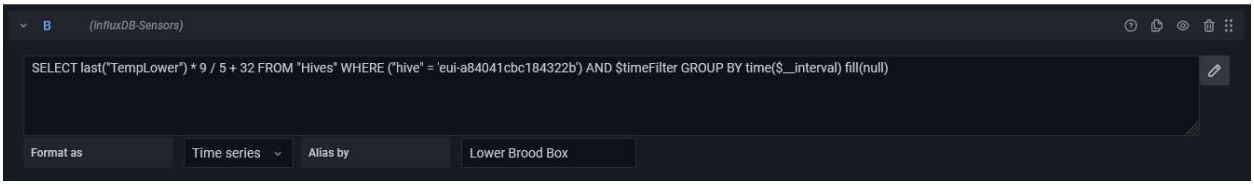


The query string is how Grafana pulls data out of the database. Dragino sensors report temperature in Celsius, so we need to do some math as we pull it out of the database to convert it to Fahrenheit. We also need to specific the EUI of the device that is on the hive.

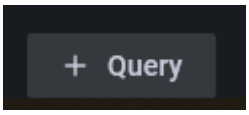
Copy and paste this text into the query builder box, updating the device EUI value.

```
SELECT last("TempLower") * 9 / 5 + 32 FROM "Hives" WHERE ("hive" = 'eui-a84041cbc184322b') AND $timeFilter GROUP BY time($__interval) fill(null)
```

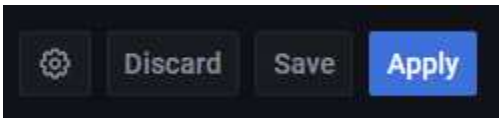
In the Alias by field you can put a friendly name of where the sensor is located. In this case “Lower Brood Box”.



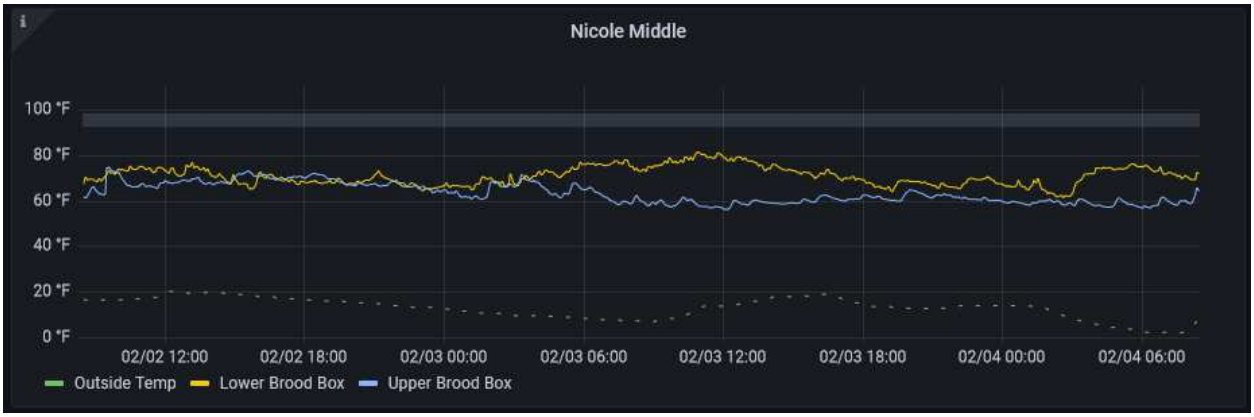
You can click on the “+ Query” to add a 2nd line to the graph if you have another temp sensor in the same hive you want to add.



Once complete, click Save and then Apply.



You will start to see something like this:



You can create additional panels for each hive.

That’s It!!

You just built your own bee hive monitoring system with real time data.

Future Appendixes

- **Importing local weather with OpenWeather API**
- **Setting up Alerts for events.**
- **Integrating Yolink temperature sensors.**
- **Integrating a scale.**
- **Creating a dashboard with sensor status (Battery, RSSI, SNR)**

Version 1.1 – February 5th, 2022

Product Links

Dragino LIG16 Indoor Gateway

https://www.robotshop.com/en/dragino-indoor-lorawan-gateway-lig16-us---915.html

Dragino DLOS8 Outdoor Antenna

https://www.robotshop.com/en/dragino-dlos8-outdoor-lorawan-gateway-915-mhz.html

Dragino LSN50v2 Sensor

https://www.robotshop.com/en/lsn50-v2-waterproof-long-range-wireless-lora-sensor-node-915-mhz.html

(The above sensor does not include temperature sensor probes.)

DS18B20 Temperature Probes

https://www.amazon.com/Aideepen-DS18B20-Waterproof-Temperature-Stainless/dp/B01LY53CED

Temperature & Humidity Probe

https://www.robotshop.com/en/sht20-i2c-temperature--humidity-sensor-waterproof-probe.html

There are a lot of other LoRa-based sensors and gateways. If they can be incorporated into TTN, you can get them incorporated.

Version 1.1 – February 5th, 2022

Dragino LSN50v2 Wiring & Configuration

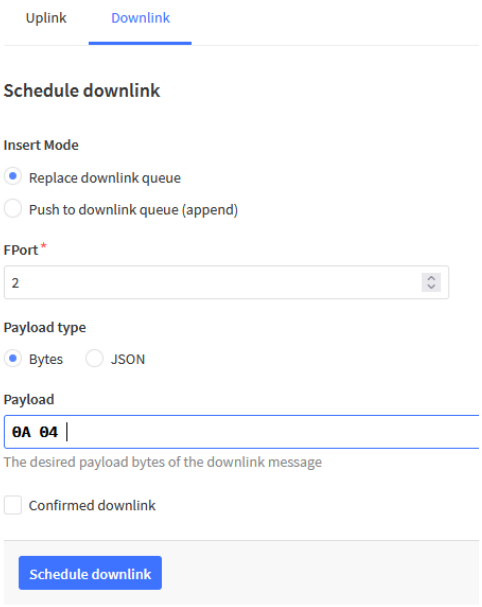
***Important* – Do not power on the unit without an antenna attached.**

Setting the working mode (MOD)

The LSN50 supports several different "working modes" referred to as 'MOD'. MOD=1 is the default mode and is configured for (1) DS18B20 sensor and some other sensor types. In order to support multiple DS18B20 sensors, we need to change it to MOD=4. There are two methods to making a change to the unit. You can buy a USB-Serial dongle and use some AT commands (blech!). The easier option is to send a Downlink payload to the device through TTN.

Log into TTN. Go to Applications → "Your application" → End Devices → Click on the device you want to change. → Click on the "Messaging" tab. → Click the "Downlink" tab.

- Set FPort = 2
- In the Payload box, type "0a04". (0a is the MOD setting, 04 sets it to MOD 4)
- Click "Schedule downlink".



The command will be sent to the device the next time it reports in. After the command is received, the device will restart in MOD=4 mode.

Modifying the transmit interval

By default, the LSN50v2 will send data every 10 minutes. If you want to shorten or lengthen that time frame, follow the instructions above for scheduling a Downlink message and use the following payload:

01xxxxxx where xxxxxx is the HEX value of the number of seconds.

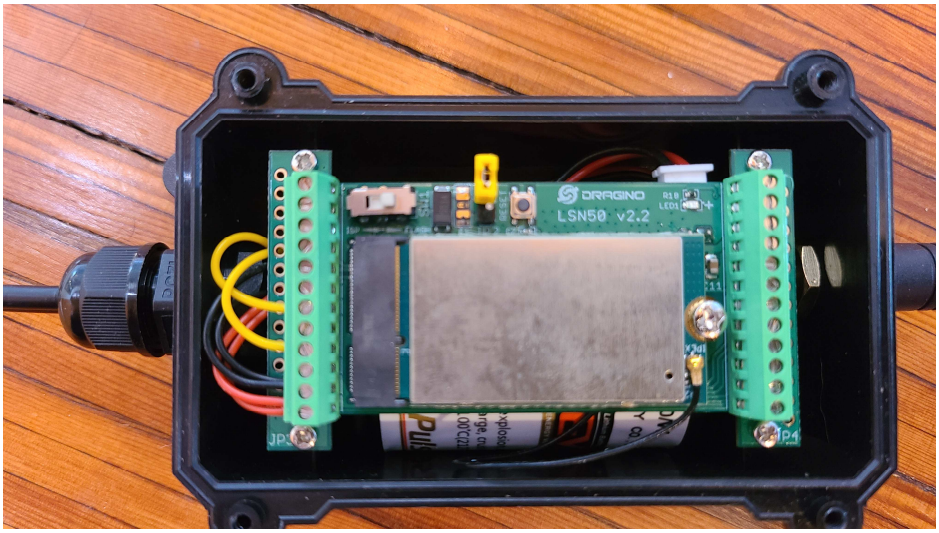
0100012c = 300 secs (5 mins)

01000258 = 600 secs (10 mins)

01000384 = 900 sec (15 mins)

Attaching the temperature probes

Here is how the wires are connected. It will make your life easier if you solder the red pairs and black pairs together before inserting them. (Of course you could also just solder them into the holes.) Red is VDD, Black is Ground and the Yellow's are the signal wires. Refer to Dragino documentation.



Two DS18B20 temperature probes attached.

Finished unit ready for installation. (Antenna upgraded)