How to compile the LoRa gateway software for the LoRa gateway HAT

In this tutorial we'll be going over how to compile the LoRa Gateway Software our LoRa Gateway HAT.

This tutorial is targeted towards the Raspberry Pi, however you can also use this as the basis for other SBCs. We also recommend on the Pi that Raspbian is used and cannot provide support for other operating systems.

This guide is for advanced users only!

This guide is targeted towards advanced users who wish to have more flexibility than our pre-done image. If you would prefer a ready to go solution please follow our guide **here**.

Doing it this method means you will not get use of our Web Interface, automtic updates and many other extra features and instead is just for the basic software to act as a LoRa Gateway.

We also assume that you can do some of the basics such as having your desired OS already written to an SD card, and that you are SSHd in or have a terminal open.

While support is limited, if you have any issues let us know by opening an issue at https://github.com/PiSupply/iotlorarange

Let's begin! - Prelim setup

Start by ensuring SPI is enabled, on the Raspberry Pi you can run sudo raspi-config select the interfacing options, then SPI, and then selecting yes. If you are on a different SBC you will need to find the guide on how to enable it for that SBC.

If you run ls /dev on the Raspberry Pi you should see spidev0.0 show up. If you're on a different SBC this number may be different so you will need to find which number it is.

On the Raspberry Pi you must change this value

On the Model 3 and above the core clock must be slowed down for the module to work, to do this open the /boot/config.txt file for editing with sudo nano /boot/config.txt

Move to the end of the file and then add the line core_freq=250 save with Ctrl-x, Y, then enter. Then reboot the system with sudo reboot.

And then re-login.

Next let's install all of the software we need, first we'll update the OS and then install all the packages with this command.

```
sudo apt -y update && sudo apt upgrade -y
```

And then run

sudo apt -y install protobuf-compiler libprotobuf-dev libprotoc-dev automake libtool autoconf git pkg-config protobuf-c-compiler libprotobuf-c-dev build-essential libc6-dev

If you are not using a debian based OS the above command & packages will need changing to the package manager you are using.

Next we want to git clone all of the repositories we need to compile the software. Copy and paste the following commands to do so.

```
git clone https://github.com/PiSupply/lora_gateway.git git clone https://github.com/PiSupply/paho.mqtt.embedded-c.git git clone https://github.com/PiSupply/ttn-gateway-connector.git git clone https://github.com/PiSupply/packet_forwarder.git
```

Next we need to go into these directories one by one and compile the software inside.

Compile Time!

lora_gateway

We recommend that you compile these in the same order as listed here.

First cd into the lora_gateway folder, and then cd into the libloragw folder.

In here we need to edit one of the files, using nano library.cfg we can bring up an editor for the configuration file. On the line that says "PLATFORM=" we want to change it from "iotloragw_fail" to "iotloragw_rpi". If you are using the Asus tinkerboard setting this to iotloragw_tnk should work but is not gauranteed.

Save and quit in nano by pressing Ctrl+x, the Y key, then enter.

If you are using the Raspberry Pi this is all you should need to do, if you are using a different SBC then you will have to edit the file in the inc folder with the same name as the line set above, in this look for the SPI_DEV_PATH line and configure this to the correct SPI Bus for your SBC.

Finally run make -j 4 to compile the software. You then don't need to do anything else so can return to the home directory with cd.

paho.mqtt.embedded-c

Next we need to compile one of the libraries, move into this library by typing cd paho.mqtt.embedded-c. No other configuration is required so we can compile it by running make -j 4.

This library then needs installing, which we can do with sudo make install

Once again return to the home directory to compile the next library.

ttn-gateway-connector

Now onto the next library, once again cd into the directory with cd ttn-gateway-connector

Next we need to copy the configuration file by running cp config.mk.in config.mk and then we can compile with make -j 4

This should compile and be ready to be installed. To "install" this one we simply need to copy the file with sudo cp bin/libttn-gateway-connector.so /usr/lib

And once again return back to the home directory to compile the last piece.

packet_forwarder

Finally we're at the last bit of software. cd into the packet forwarder by running cd packet_forwarder and then the sub folder mp_pkt_fwd with cd mp_pkt_fwd.

Then once again compile with make -j 4.

And then we're all compiled!

Preparing to run the software

Next we're going to move all the files we need to one directory. This has to be /opt/iotloragateway.

So to do this we're going to create the folder with sudo mkdir /opt/iotloragateway and change the permissions so we can write and execute the files inside with sudo chmod 777 /opt/iotloragateway.

Next from inside the mp_pkt_fwd folder we're in we need to copy just the "mp_pkt_fwd" file to the new directory with cp mp_pkt_fwd /opt/iotloragateway/

Next move to that directory with cd /opt/iotloragateway. Now we're going to download the configuration templates.

First we'll download the local_conf template, this is simply done with

wget
https://raw.githubusercontent.com/PiSupply/iot-lora-gw-pktfwd/master/lora_templates/local_conf.json.template

We then need to rename this to remove the template from the end by using mv local_conf.json.template local_conf.json

Next we need to download the frequency plan, you may require to find this elsewhere but we have many templates available for different regions on our repository at https://github.com/PiSupply/iot-lora-gw-pktfwd/tree/master/lora_templates

Click the global_conf template for the region you require (In our case we'll be using the EU plan) and then click raw. This should bring up the file on it's own in the browser. Copy this link (In my case https://raw.githubusercontent.com/PiSupply/iot-lora-gw-pktfwd/master/lora_templates/EU-global_conf.json and download it with wget. The full command will be like wget https://raw.githubusercontent.com/PiSupply/iot-lora-gw-pktfwd/master/lora_templates/EU-global_conf.json (You can also just change the EU to the region you require in this command).

Next we just need to rename the file to remove the region at the beginning, in my case this is done with mv EU-global_conf.json global_conf.json.

Finally we need to download the script to reset the module each time, you can do this by running wget https://raw.githubusercontent.com/PiSupply/iot-lora-gw-pktfwd/master/files/reset-22.sh

After downloading run chmod +x reset-22.sh

On the Raspberry Pi our hat uses BCM22 which is what this script is set to reset, other SBCs will be on different pin numbers so you will have to edit it for other boards.

Configuration File Setup - TTN

Next we need to edit multiple lines in the configuration file, for this we'll assume you've already setup your gateway on the TTN Console already. Primarily do not check that you are using Legacy Mode.

If you do not want to use TTN skip to the next section.

Now open the local_conf.json file with your favourite text editor, I'll be using nano.

Move to the first configurable section inbetween the quote marks next to gateway_ID, this can be anything you like and while it doesn't have to match the ID you set in the TTN Console it can be, so I'll be using this.

Next for your email simply enter your email, and then for description enter a friendly description of your choosing in.

Next move to serv_type, we need to set this to ttn. For server address you need to type in the address for the TTN router you selected, currently this is typically "bridge..thethings.network" so in my case it's "bridge.eu.thethings.network".

Next for the serv_gw_id copy the ID exactly as it shows on TTN. Then for the serv_gw_key copy and paste the gateway key from the TTN Console.

Next move down to GPS, if you have a GPS Module connected then leave this as true and change the fake_gps to false, if you don't have a GPS module connected change this to false and leave the fake_gps as true.

If you wish to have the location of your gateway display you will need to change the latitude, longitude and altitude of the gateway to where you are. Using a site such as https://www.latlong.net/ can help you do this. These do not require quote marks around the numbers.

Finally if you are using the GPS module change the serial port to the correct address, on the RPi this is /dev/serial0 so requries changing from the default.

Here's a screenshot of what it should look like, the key will be longer but I have trimmed the screenshot as the key should not be shared.

Finally save with ctrl-x, y then enter,

Configuration File Setup - Legacy

This will be written soon. If you have set your gateway up with TTN as above must skip this section.

Testing

Finally we can test to see if we've done it right. first reset the module by running sudo ./reset-22.sh. and then Launch the packet forwarder by running ./mp_pkt_fwd

If all has worked then you should get a screen like the following:

```
pi@raspberrypi:/opt/iotloragateway $ ./mp_pkt_fwd
14:04:50  *** Multi Protocol Packet Forwarder for Lora Gateway ***
Version: 3.0.20
14:04:50  *** Lora concentrator HAL library version info ***
Version: 5.0.1; Options: native;
***
14:04:50  INFO: Little endian host
14:04:50  INFO: found global configuration file /opt/iotloragateway/global_conf.json, parsing it
14:04:50  INFO: /opt/iotloragateway/global_conf.json does contain a JSON object named SX1301_conf, parsing SX1301 parameters
14:04:50  INFO: lorawan_public 1, clksrc 1
14:04:50  INFO: no configuration for LBT
14:04:50  INFO: antenna_gain 0 dBi
14:04:50  INFO: Configuring TX LUT with 16 indexes
14:04:50  INFO: radio 0 enabled (type SX1257), center frequency 867500000, RSSI offset -166.000000, tx enabled 1
14:04:50  INFO: radio 1 enabled (type SX1257), center frequency 868500000, RSSI offset -166.000000, tx enabled 0
14:04:50  INFO: Lora multi-SF channel 0>  radio 1, IF -400000 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora multi-SF channel 1>  radio 1, IF -200000 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora multi-SF channel 2>  radio 1, IF 0 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora multi-SF channel 3>  radio 0, IF -400000 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora multi-SF channel 4>  radio 0, IF -200000 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora multi-SF channel 5>  radio 0, IF 0 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora multi-SF channel 6>  radio 0, IF 200000 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora multi-SF channel 7>  radio 0, IF 400000 Hz, 125 kHz bw, SF 7 to 12
14:04:50  INFO: Lora std channel> radio 1, IF -200000 Hz, 250000 Hz bw, SF 7
14:04:50  INFO: FSK channel> radio 1, IF 300000 Hz, 125000 Hz bw, 50000 bps datarate
14:04:50  INFO: /opt/iotloragateway/global_conf.json does contain a JSON object named gateway_conf, parsing gateway parameters
14:04:50  INFO: Found 1 servers in array.
14:04:50  INFO: Server 0 configured to "router.eu.thethings.network"
14:04:50  INFO: packets received with a valid CRC will be forwarded
14:04:50  INFO: packets received with a CRC error will NOT be forwarded
14:04:50  INFO: packets received with no CRC will NOT be forwarded
14:04:50  INFO: GPS is disabled
14:04:50  INFO: Upstream data is enabled
14:04:50  INFO: Downstream data is enabled
14:04:50  INFO: Ghoststream data is disabled
14:04:50  INFO: Radiostream data is enabled
14:04:50  INFO: Statusstream data is enabled
14:04:50  INFO: Beacon is disabled
14:04:50  INFO: Packet logger is disabled
14:04:50  INFO: Flush output after statistic is disabled
14:04:50  INFO: Flush after each line of output is disabled
14:04:50  INFO: Watchdog is disabled
14:04:50  INFO: found local configuration file /opt/iotloragateway/local_conf.json, parsing it
14:04:50  INFO: redefined parameters will overwrite global parameters
14:04:50  INFO: /opt/iotloragateway/local_conf.json does not contain a JSON object named SX1301_conf
14:04:50  INFO: /opt/iotloragateway/local_conf.json does contain a JSON object named gateway_conf, parsing gateway parameters
14:04:50  INFO: gateway MAC address is configured to 0000000000000000
14:04:50  INFO: Found 1 servers in array.
14:04:50  INFO: Server 0 configured to "bridge.eu.thethings.network"
14:04:50  INFO: packets received with a valid CRC will be forwarded
14:04:50  INFO: packets received with a CRC error will NOT be forwarded
14:04:50  INFO: packets received with no CRC will NOT be forwarded
14:04:50  INFO: GPS serial port path is configured to "/dev/serial0"
14:04:50  INFO: Reference latitude is configured to 52.557240 deg
14:04:50  INFO: Reference longitude is configured to 1.720660 deg
```

Quit the packet forwarder by pressing ctrl+c at the same time.

Start on boot

Finally we'll add a service to start the service on boot.

Start by downloading the script usng wget
https://raw.githubusercontent.com/PiSupply/IoTLoRaRange/master/IoT%20Lora%20Gateway%20HAT/scripts/iot-lora-gateway.service

Then copy the file to the correct directory using sudo install -m 644 iot-lora-gateway.service /lib/systemd/system/

And then finally run the following to setup the script

```
sudo systemctl daemon-reload sudo systemctl enable iot-lora-gateway.service
```

Then reboot the system and the software should start automatically and you're complete!