

# MQTT Forward Instruction

From Wiki for Dragino Project

## Contents

- 1 INTRODUCTION
  - 1.1 Support Devices
- 2 Firmware Change Log for MQTT feature
- 3 MQTT forward operating principle
  - 3.1 Network Structure
  - 3.2 How sensor data is forwarded
    - 3.2.1 Upstream
    - 3.2.2 Downstream
  - 3.3 Macro Definition
    - 3.3.1 -t topic macro
    - 3.3.2 -m message macro
    - 3.3.3 Example for Macro
- 4 Example to communicate to a simple MQTT server
  - 4.1 Overview
  - 4.2 Simulate via MQTT.fx utility
  - 4.3 Simulate via Dragino Command Line
  - 4.4 Configure Dragino UI for MQTT connection
    - 4.4.1 Configure the MQTT Client for Upstream
    - 4.4.2 Configure the MQTT Client for Downstream
  - 4.5 Add LoRa support to communicate with remote sensor
    - 4.5.1 Use LoRa Raw protocol for communication -- For LG01/LG02
    - 4.5.2 Use LoRaWAN Protocol for communication -- For LG308/LPS8/DLOS8
- 5 Example For Different MQTT Servers
  - 5.1 ThingSpeak Server
  - 5.2 乐联网平台
  - 5.3 AWS-IOT

## INTRODUCTION

Dragino LoRa/LoRaWAN gateway support MQTT forwarding. It can forward the sensor data from LoRa network to MQTT server , and vice verse.

## Support Devices

This MQTT forward instruction is for below devices:

- Firmware Version > LG02\_LG08-5.3.1580178039 Firmware Download  
([http://www.dragino.com/downloads/index.php?dir=LoRa\\_Gateway/LPS8/Firmware/Release/](http://www.dragino.com/downloads/index.php?dir=LoRa_Gateway/LPS8/Firmware/Release/))
- LG01N, OLG01N (Warning: LG01-P LG01-S use another instruction: MQTT for LG01-P/LG01S)
- LG02, OLG02
- LG308, DLOS8
- LPS8
- MS14 series if installed with the same firmware. (in this case, the MQTT forward will work , but no LoRa support)

# Firmware Change Log for MQTT feature

This instruction is wrote start from LG02\_LG08-5.3.1580178039. Below is related change log since this version of firmware.

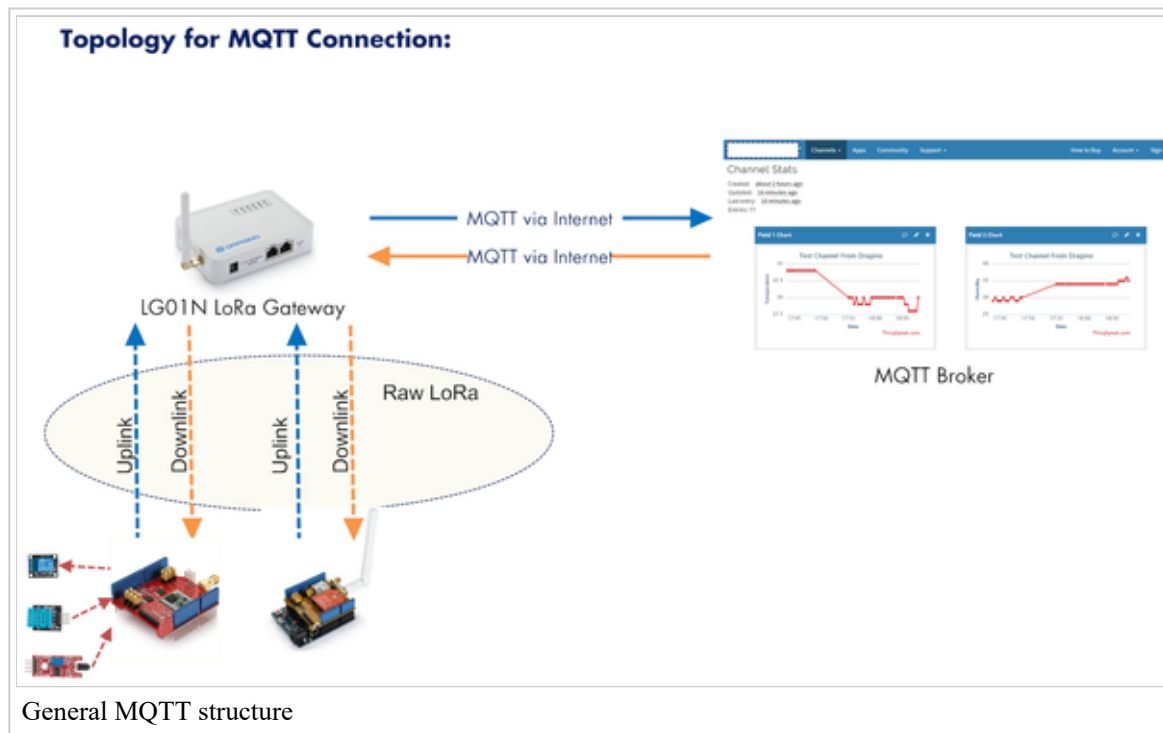
- LG02\_LG08-5.3.1580178039
  - Initiate version

## MQTT forward operating principle

### Network Structure

Below shows the network structure for MQTT forwarding.

- For Uplink: The sensor sends data to LoRa Gateway via LoRa wireless, The gateway will process these data and forward to remote MQTT Broker via Internet.
- For Downlink: The gateway subscribe a topic in the MQTT broker, when there is update on the topic, the gateway will know and broadcast the data to Local LoRa network,



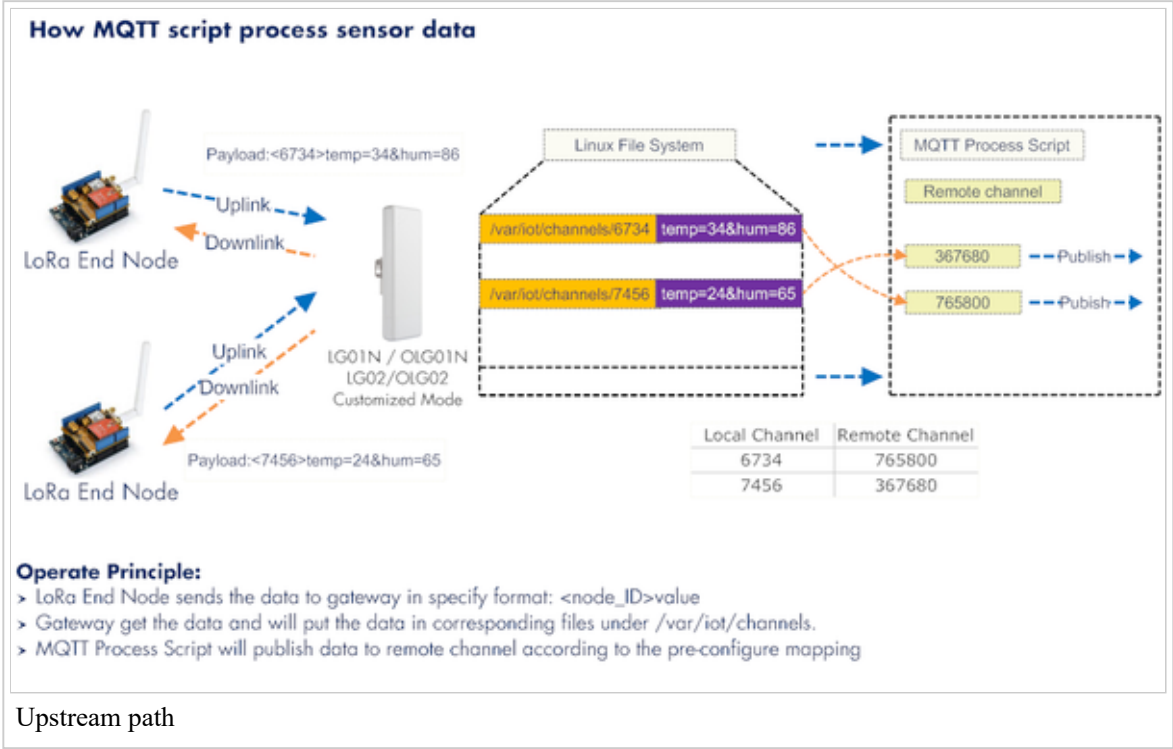
## How sensor data is forwarded

In this MQTT forward feature, the key point is how the gateway process the sensor data.

### Upstream

Assume there are two sensor nodes, their ID are Node1 ID: 6734 , Node2 ID: 7456. In the remote MQTT broker there are two topics: Topic1: /channel/765800, Topic2: /channel/367860. We can set up in the gateway to map Node1 to Topic1 and Node2 to Topic2. So when there is a sensor data from Node1, the gateway will forward the data to Topic1, when there is sensor data from Node2, the gateway will forward to Topic2.

The data flow works as below diagram.

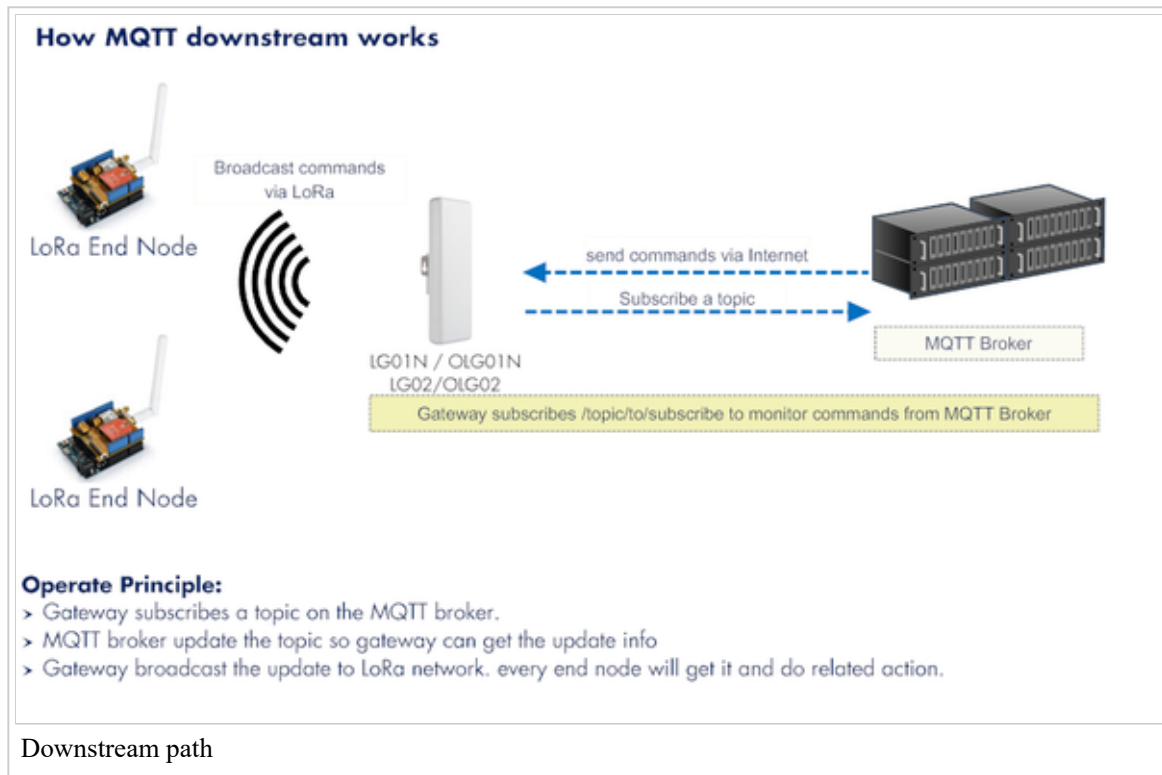


Note: The sensor data can base on LoRa or other method, as long as there are data on the file /var/iot/channels.

Downstream

The gateway subscribes to a topic of the remote MQTT broker topic. When there is some one publish a value on this topic. The gateway will get it and broadcast to local LoRa Network.

Below are the data flow for downstream.



## Macro Definition

The MQTT publish command use Macro settings to generate flexible upstream payload for MQTT publish.

Currently the -t (topic) and -m (message) support Macros.

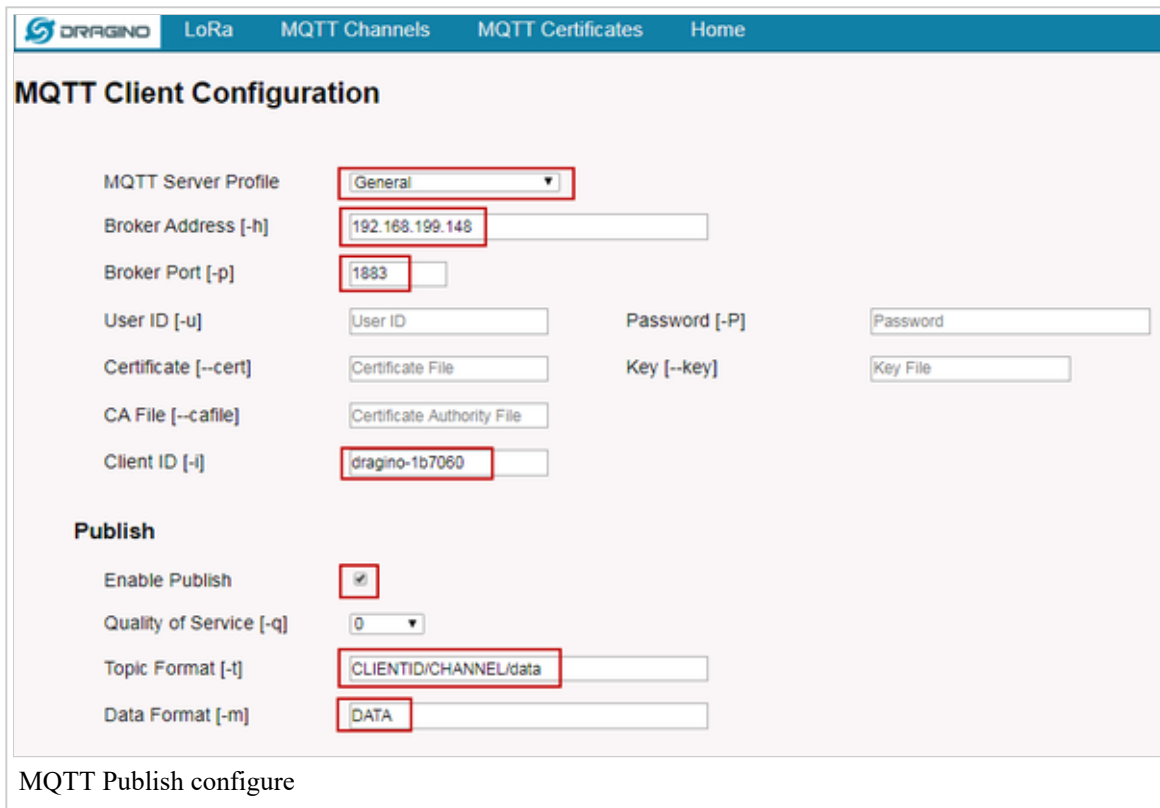
### -t topic macro

- CHANNEL: Remote Channel ID
- CLIENTID: Client ID , Same as -i
- WRITE\_API: Remote Channel Write API
- USERNAME: User ID (-u)
- HOSTNAME: Device Hostname

### -m message macro

- HOSTNAME: Device Hostname
- CHANNEL: Remote Channel ID
- DATA: Sensor Data without time stamp and rssi
- META: Completely sensor data with time stamp and rssi
- JSON: Convert META to json format.

### Example for Macro



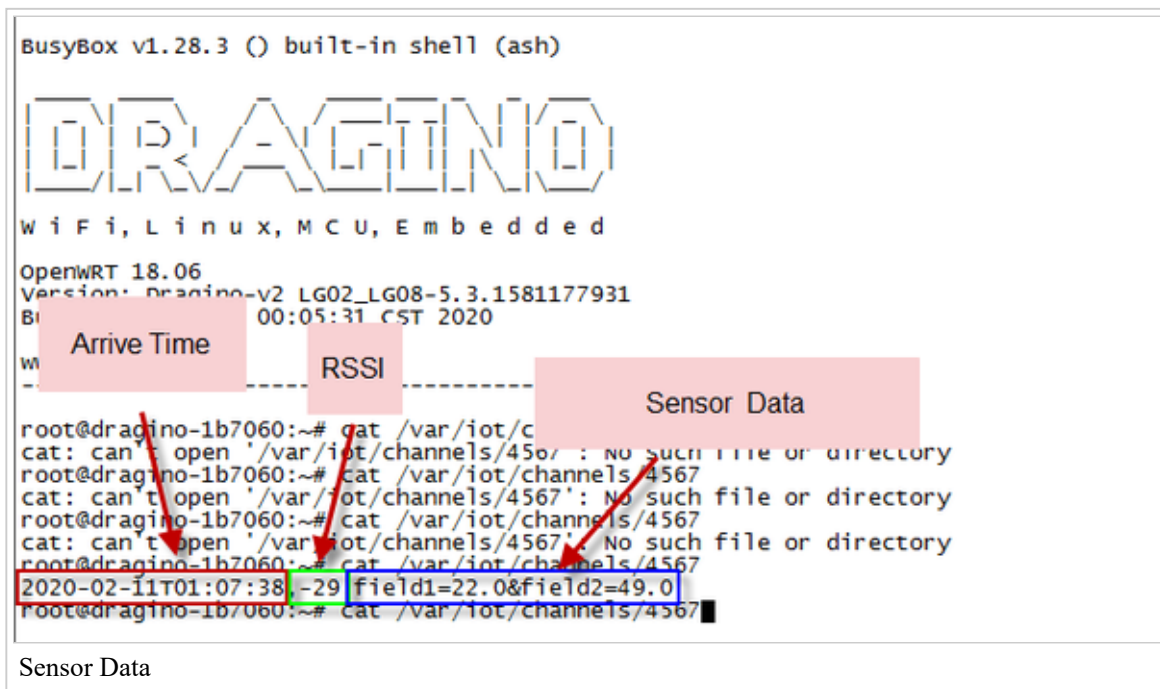
The image shows the 'MQTT Client Configuration' web interface. It has a navigation bar with 'LoRa', 'MQTT Channels', 'MQTT Certificates', and 'Home'. The main section is titled 'MQTT Client Configuration'. It contains several input fields and a 'Publish' section. The 'MQTT Server Profile' is set to 'General'. The 'Broker Address [-h]' is '192.168.199.148'. The 'Broker Port [-p]' is '1883'. The 'User ID [-u]' is 'User ID' and the 'Password [-P]' is 'Password'. The 'Certificate [--cert]' is 'Certificate File' and the 'Key [--key]' is 'Key File'. The 'CA File [--cafile]' is 'Certificate Authority File'. The 'Client ID [-i]' is 'dragino-1b7060'. The 'Publish' section has 'Enable Publish' checked, 'Quality of Service [-q]' set to '0', 'Topic Format [-t]' set to 'CLIENTID/CHANNEL/data', and 'Data Format [-m]' set to 'DATA'.

MQTT Publish configure

Above screen shots shows below format:

- -t: CLIENTID/CHANNEL/data
- -m: DATA

When there is a LoRa sensor arrive. it will be store at the /var/iot/channels as below:



The image shows a terminal window with the title 'BusyBox v1.28.3 () built-in shell (ash)'. It displays the 'DRAGINO' logo and system information: 'W i f i , L i n u x , M C U , E m b e d d e d', 'OpenWRT 18.06', 'Version: Dragino-v2 LG02\_LG08-5.3.1581177931', and 'Build Date: 2020-02-11T01:07:38 CST 2020'. Below this, there is a table with columns 'Arrive Time', 'RSSI', and 'Sensor Data'. The table shows a single row of data: '2020-02-11T01:07:38 -29 field1=22.0&field2=49.0'. The 'Sensor Data' column is highlighted with a red box. The terminal prompt is 'root@dragino-1b7060:~#'. Below the table, there are several lines of error messages: 'cat: can't open '/var/iot/channels/4567': No such file or directory', 'cat: can't open '/var/iot/channels/4567': No such file or directory', 'cat: can't open '/var/iot/channels/4567': No such file or directory', and 'cat: can't open '/var/iot/channels/4567': No such file or directory'.

Sensor Data

According to above macro. Gateway will publish field1=22.0&field2=49.0 to topic: dragino-1b7060/78901/data, where 78901 is the remote channel for this node ID.

# Example to communicate to a simple MQTT server

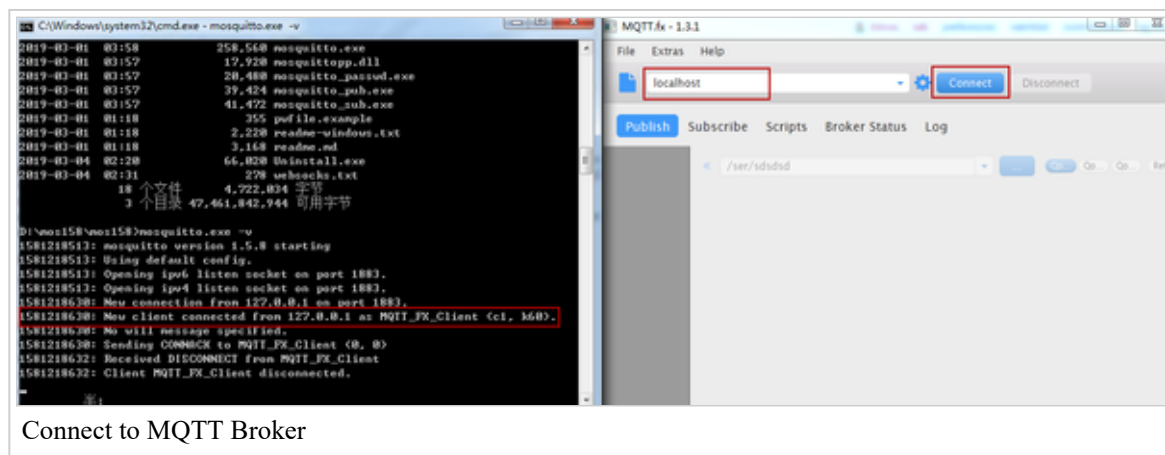
## Overview

This section is an example to show how to set up LG01-N to communicate with a MQTT server. The MQTT server is a simple utility set up in a local PC. Note: User can set up same server via this instruction (<http://www.steves-internet-guide.com/install-mosquitto-broker/>).

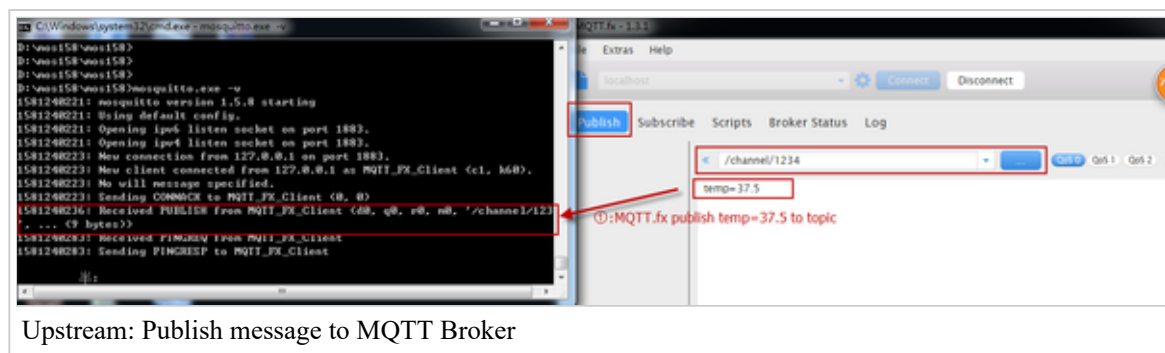
## Simulate via MQTT.fx utility

The MQTT.fx (<http://mqttfx.jensd.de/index.php/download>) is a MQTT client tool. We can use this to simulate a MQTT connection to our MQTT broker first to make sure the MQTT broker works. This will also help us understand how it works.

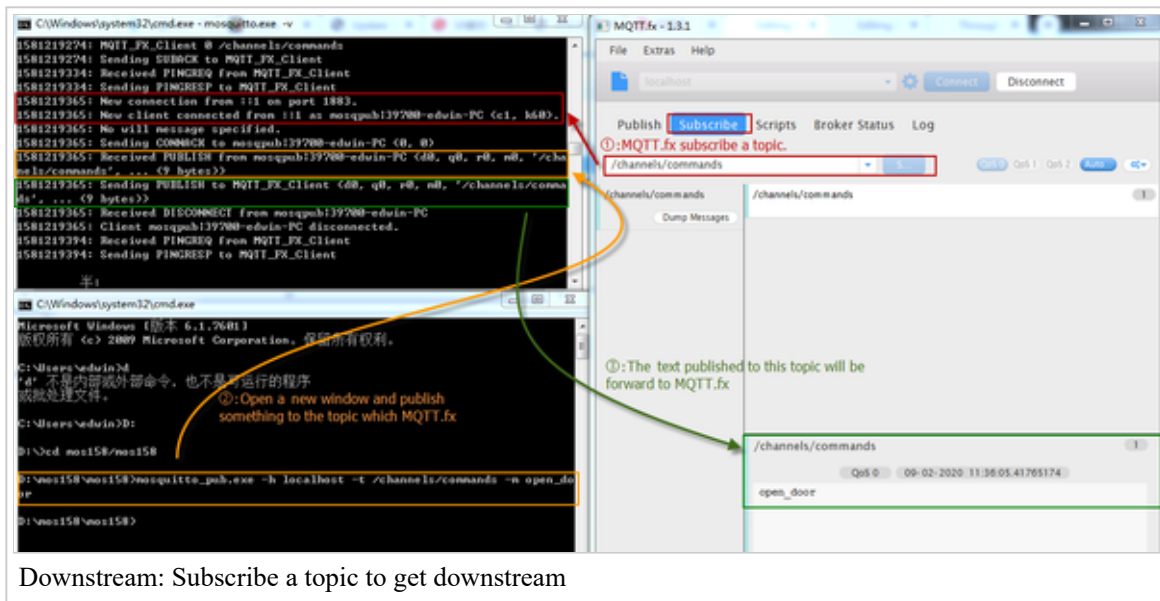
In this test, the MQTT broker and MQTT.fx are installed in the same PC, so the MQTT server address in MQTT.fx should be localhost. Below shows how to connect to the server.



After connected, use publish to public some thing to MQTT server. This to simulate upstream



To simulate a downstream, use MQTT.fx to subscribe a topic, and publish something to this topic. as Below:



## Simulate via Dragino Command Line

For first try of MQTT connection, simulate via command line is recommend, there are many servers / connection type for MQTT. They are using different connection parameters. Simulating the connection via command line will help us rapidly connect to server and debug.

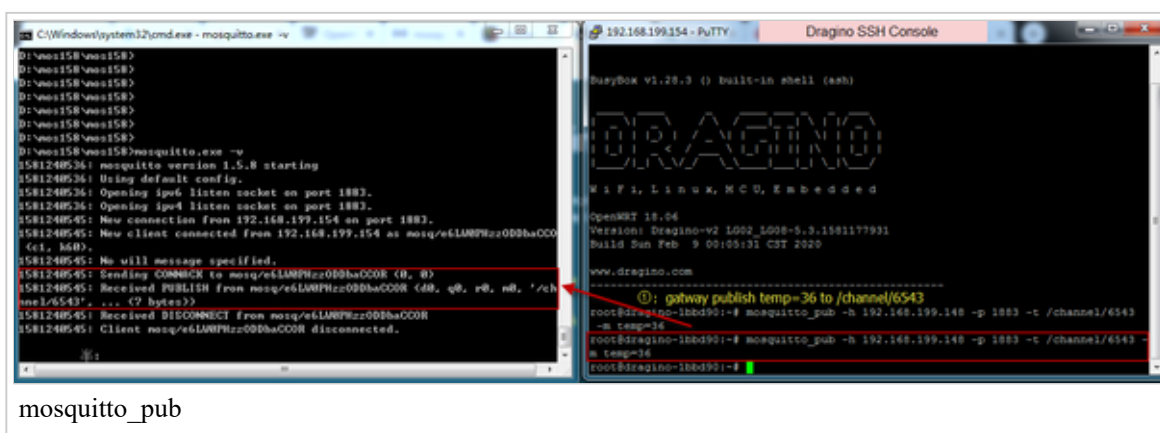
In the Dragino Gateway, we use mosquitto client (<https://mosquitto.org/>) for MQTT connection.

For Upstream

command is mosquitto\_pub ([https://mosquitto.org/man/mosquitto\\_pub-1.html](https://mosquitto.org/man/mosquitto_pub-1.html))

Example: mosquitto\_pub -h 192.168.199.148 -p 1883 -t /channel/6543 -m temp=36

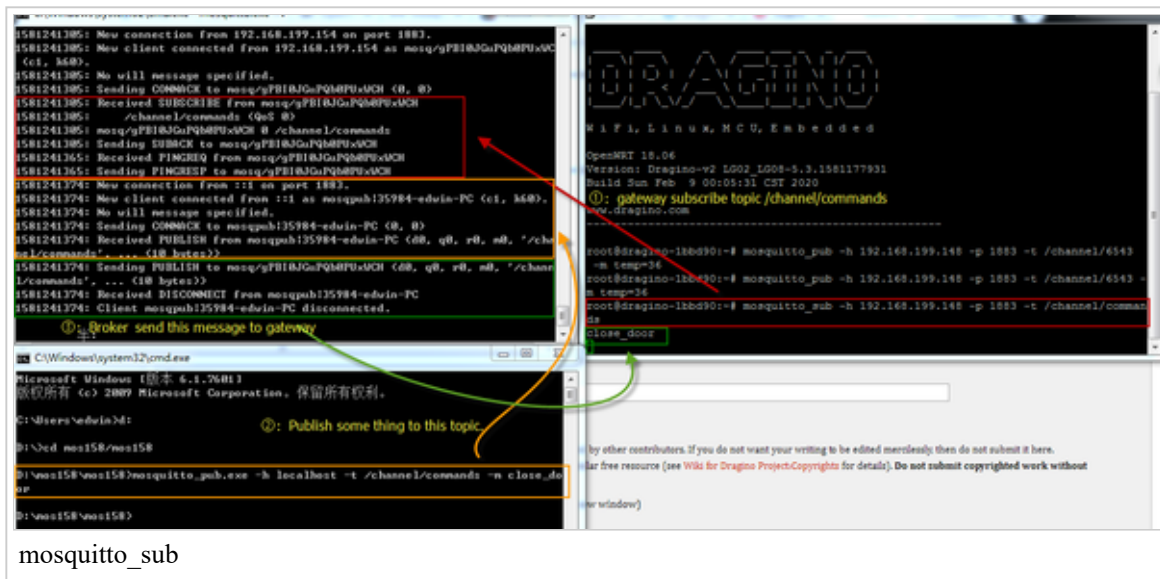
Note: 192.168.199.148 is MQTT broker address, the gateway and the MQTT broker PC are in the same network.



For Downstream

Use mosquitto\_sub ([https://mosquitto.org/man/mosquitto\\_sub-1.html](https://mosquitto.org/man/mosquitto_sub-1.html)) to subscribe the change on the topic.

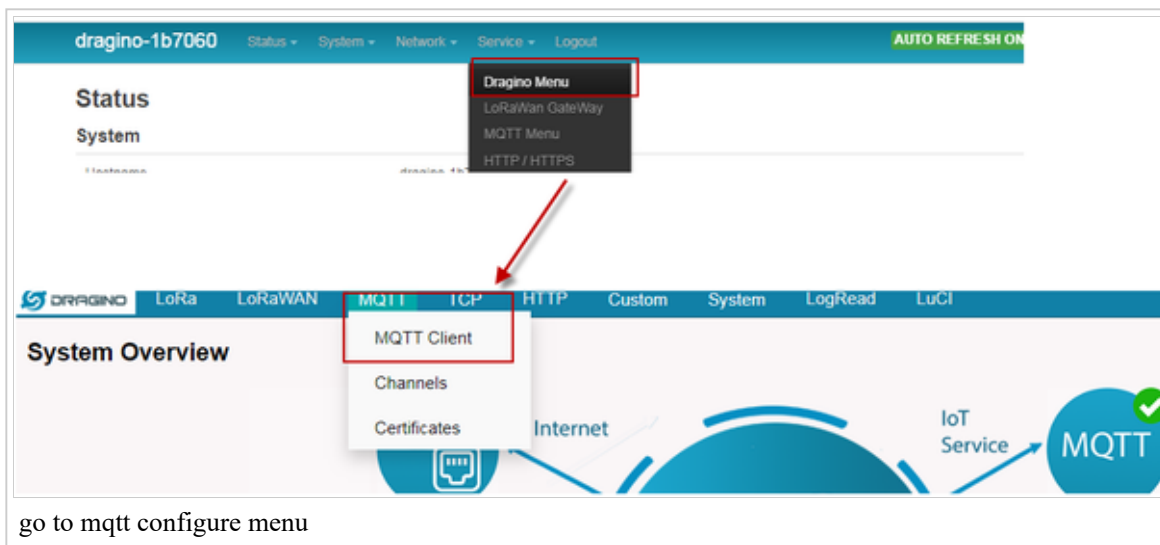




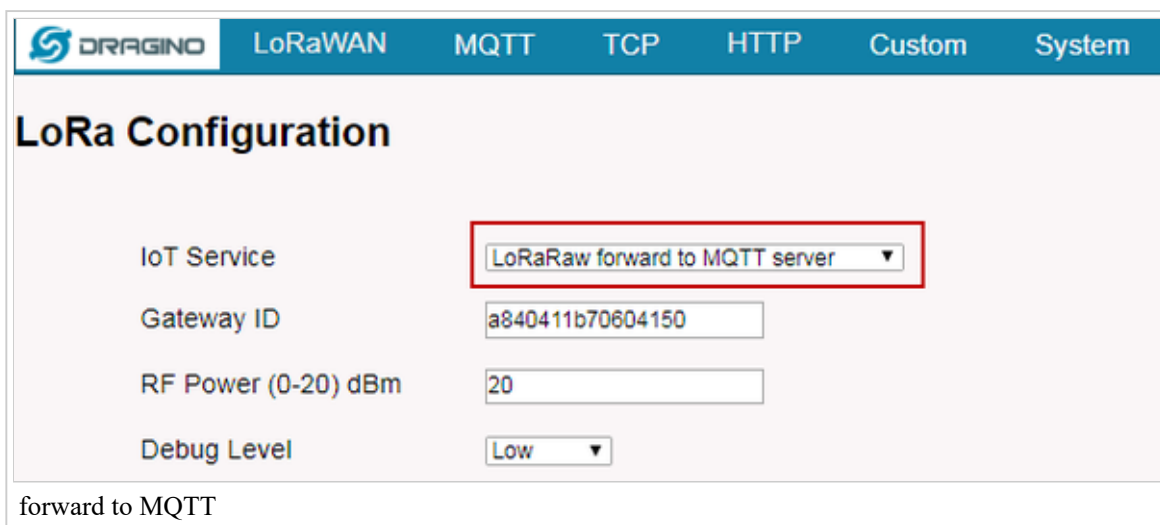
## Configure Dragino UI for MQTT connection

This chapter are step by step to show to configure the Dragino Menu for MQTT auto connection.

Go to Dragino Menu --> MQTT Client



Select Forward to MQTT server. Notice: This option is removed from the latest firmware, in the latest firmware, if user submit "SAVE & APPLY" in MQTT page, the gateway will use MQTT service.





## Configure the MQTT Client for Upstream

Below screenshot is same as the publish command:

```
mosquitto_pub -h 192.168.199.148 -p 1883 -i dragino-1b7060 -t CLIENTID/CHANNEL/data -m DATA
//where the CLIENTID, CHANNEL & DATA are macro. represent for
//CLIENTID: dragino-1b7060
//CHANNEL: Remote ID in Channel settings; here is 78901 or 567456
//DATA: The data stores in /var/iot/channels/
```

**MQTT Client Configuration**

MQTT Server Profile: General

Broker Address [-h]: 192.168.199.148

Broker Port [-p]: 1883

User ID [-u]: User ID Password [-P]: Password

Certificate [--cert]: Certificate File Key [--key]: Key File

CA File [--cafile]: Certificate Authority File

Client ID [-i]: dragino-1b7060

**Publish**


Enable Publish: ☒

Quality of Service [-q]: 0

Topic Format [-t]: CLIENTID/CHANNEL/data

Data Format [-m]: DATA

MQTT Publish configure

 [MQTT Config](#) [MQTT Certificates](#) [Home](#)

## MQTT Channel Management

### Add / Edit Channel

Channel Number

Local ID

Remote ID

Write API Key

Number 0 - 99

Add

### Delete Channel

Channel Number

Number 0 - 99

Delete

This define how the sensor node match the remote channel.

### Saved Channels:

chan1 Local ID: 4567 Remote ID: 78901 Write API Key:

chan2 Local ID: 3423 Remote ID: 567456 Write API Key:

Refresh

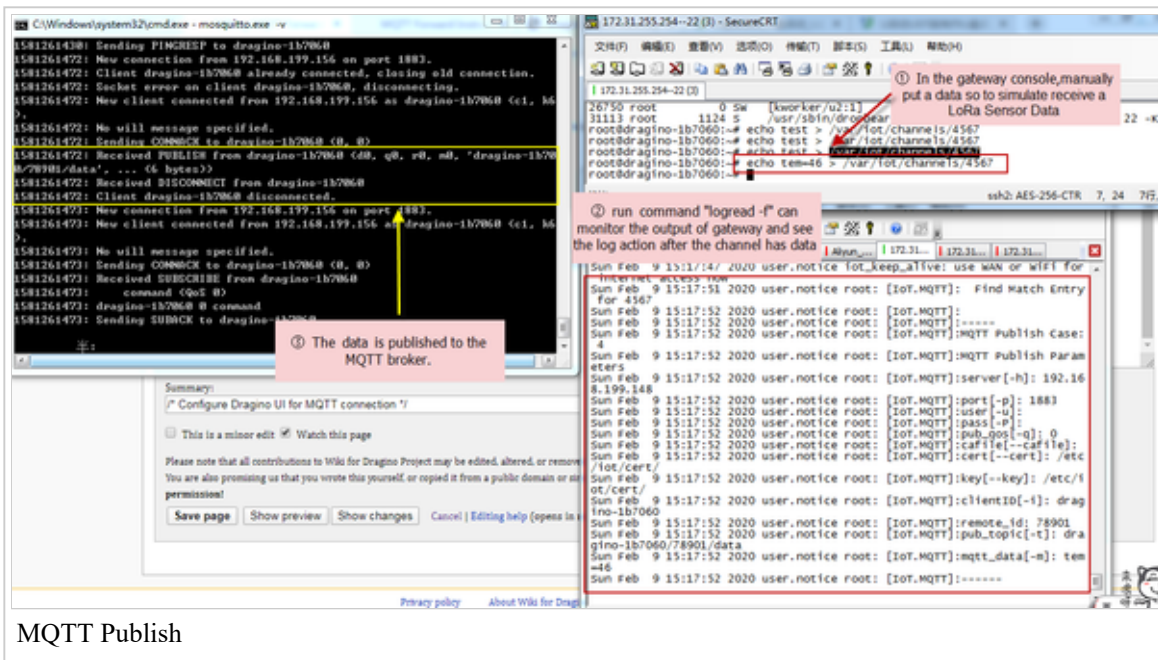
MQTT Channel settings

For example, if we put a data(temp=46) on the file /var/iot/channels/4567, because 4567 match the remote channel 78901. the gateway will run this command:

```
mosquitto_pub -h 192.168.199.148 -p 1883 -i dragino-1b7060 -t dragino-1b7060/78901/data -m temp=46
```

to MQTT broker.

Below is a simulation to put this data to active the MQTT publish.



MQTT Publish

## Configure the MQTT Client for Downstream

Below screen shot equal to this subscribe command:

```
mosquitto_sub -h 192.168.199.148 -p 1883 -i dragino-1b7060 -t command.
```

### Subscribe

Enable Subscribe

☒

Quality of Service [-q]

0 ▼

Topic Format [-t]

command

Save

Save&Apply

Cancel

MQTT Subscribe

When MQTT broker receive a update on this topic, the gateway will get the update and use LoRa radio to broadcast this message. The LoRa parameters used for update is:

**LoRa Configuration**

IoT Service:

Gateway ID:

RF Power (0-20) dBm:

Debug Level:

**Radio Settings**

Frequency (Hz):  RF Bandwidth (Hz):

Spreading Factor:  Coding Rate:

Preamble Length:  LoRa Sync Word:

LoRa Broadcast parameters.

And below is the subscribe simulation:

downstream simulation

## Add LoRa support to communicate with remote sensor

In above section, we have configured the UI to support MQTT upstream and downstream. We can simulate via Linux command. In this section, we will guide how to communicate with remote LoRa End Node for upstream and downstream.

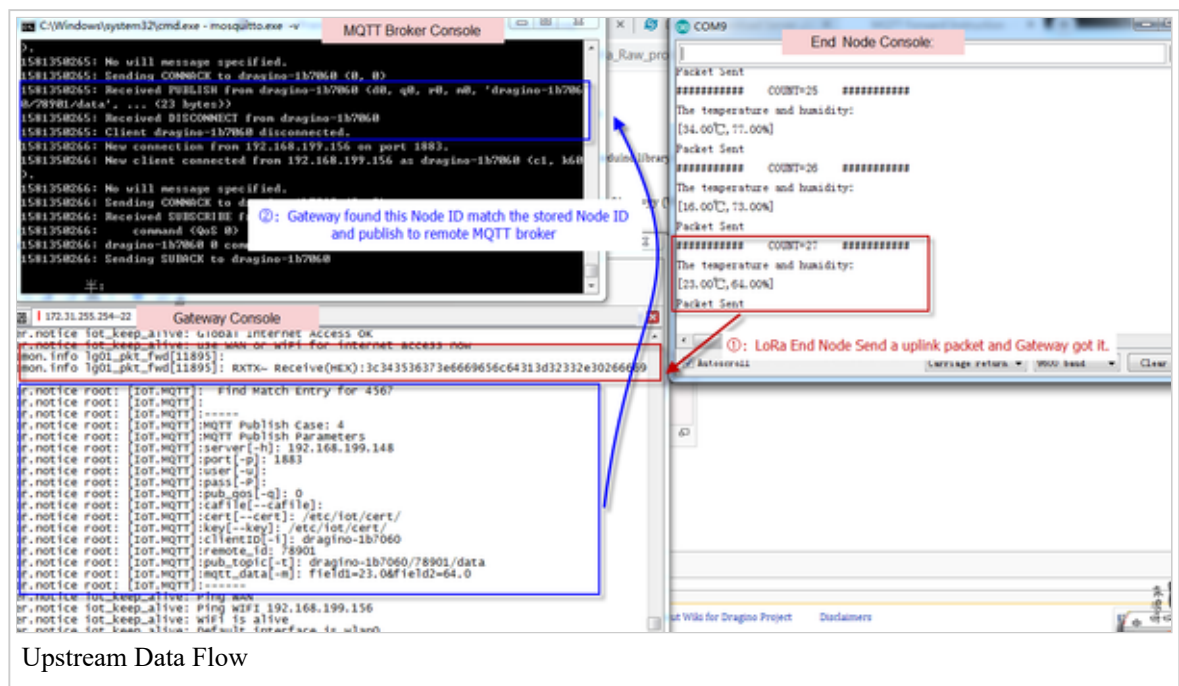
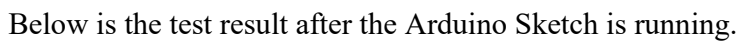
### Use LoRa Raw protocol for communication -- For LG01/LG02

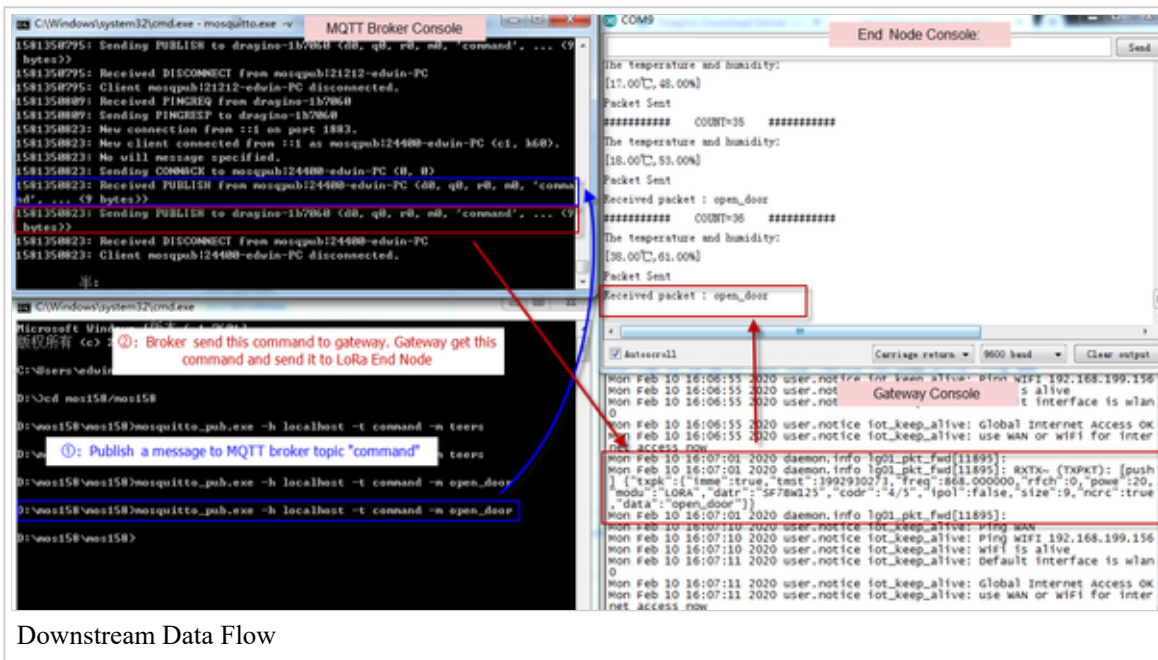
We can use LoRa Shield (<http://www.dragino.com/products/lora/item/102-lora-shield.html>) to send LoRa Raw data to Gateway and receive data from gateway.

The example Sketch for LoRa Shield +Arduino is here: [LoRa\\_Shield\\_Sketch\\_For\\_MQTT](http://www.dragino.com/downloads/index.php?dir=LoraShield/) (<http://www.dragino.com/downloads/index.php?dir=LoraShield/>)

## What does the Arduino Sketch do? The Arduino Sketch will:

- Upstream: Keep sending a LoRa Message every minutes with this payload : <4567>tem=xx&hum=yy (Where xx and yy are temperature and humidity value generated randomly).
- Downstream: Listening broadcast message from gateway, and print it in console.
- The LoRa parameter settings in Arduino should match the LoRa settings in gateway, as below:





Downstream Data Flow

## Use LoRaWAN Protocol for communication -- For LG308/LPS8/DLOS8

Since firmware LG02\_LG08--build-v5.3.1585192026-20200326-1109, Dragino LoRaWAN gateways support the communication to LoRaWAN ABP end node locally without the need of LoRaWAN server. This feature allow us to integrate MQTT in the gateway to support LoRaWAN to MQTT forwarding or visa verse.

When use test this feature, please use the version higher then : LG02\_LG08--build-v5.4.1593400722-20200629-1120, in this version, the upload format is changed and readable, which is easier for integration.

Video Instruction:<https://youtu.be/qJTY441-t90>

Step 1: Refer Communicate with ABP End Node to know how to set up LG308 to work with LoRaWAN End node.

Step 2: Make sure your Radio settings match the End Node settings.



**LoRa Configuration**  
Debug Level: Low

**Radio Settings**

Frequency Plan: United States 915Mhz(902~928) -- US915

Frequency Sub Band: 2: US915(903.9~905.3) / AU915(916.8~918.2)

Save Save&Apply Disable Cancel

**Previous Log:**

```
Thu Mar 26 03:50:48 2020 daemon.info lora_pkt_fwd[30146]: INFO~ [down] PULL_ACK received in 285 ms
Thu Mar 26 03:50:54 2020 daemon.info lora_pkt_fwd[30146]: INFO~ [down] PULL_ACK received in 295 ms
Thu Mar 26 03:50:59 2020 daemon.info lora_pkt_fwd[30146]: INFO~ [down] PULL_ACK received in 295 ms
Thu Mar 26 03:51:02 2020 user.notice iot_keep_alive: Ping WAN 10.130.2.189
Thu Mar 26 03:51:04 2020 daemon.info lora_pkt_fwd[30146]: INFO~ [down] PULL_ACK received in 295 ms
Thu Mar 26 03:51:06 2020 user.notice iot_keep_alive: Ping WIFI
Thu Mar 26 03:51:06 2020 user.notice iot_keep_alive: Default interface is 3g-cellular
Thu Mar 26 03:51:09 2020 daemon.info lora_pkt_fwd[30146]: INFO~ [down] PULL_ACK received in 295 ms
Thu Mar 26 03:51:10 2020 daemon.info lora_pkt_fwd[30146]: DATA_UNCONF_UP:{"DevAddr": "2602111D", "FCtrl": [{"ADR": 1, "ADRACKReq": 0, "ACK": 0, "RFU": "RFU", "FOptsLen": 0}, {"FCnt": 57, "FPort": 2, "MIC": "449C3D13"}]}
Thu Mar 26 03:51:10 2020 daemon.info lora_pkt_fwd[30146]: DATA_UNCONF_UP:{"DevAddr": "2602111D", "FCtrl": [{"ADR": 1, "ADRACKReq": 0, "ACK": 0, "RFU": "RFU", "FOptsLen": 0}, {"FCnt": 57, "FPort": 2, "MIC": "449C3D13"}]}
Thu Mar 26 03:51:10 2020 daemon.info lora_pkt_fwd[30146]: RXTX~ [{"rxpk": [{"tmst": 275900532, "time": "2020-03-26T03:51:10.221253Z", "chan": "S", "rfch": "1", "freq": 904.900000, "stat": "1", "modu": "LORA", "datr": "SF10BW125", "codr": "4/5", "lsnr": -12.5, "rssi": -88, "size": 24, "data": "QB0RAiaAQ ACg1PHdw0jzpeNuoTPZME"}]]
```

Use Same Frequency Band as End Node

Step 3: Set up publish format and MQTT channel. The LG308 will store the Data from End node in HEX format in the file. And we need to config the format to META

**MQTT Client Configuration**

MQTT Server Profile: General

Broker Address [-h]: 10.130.2.192

Broker Port [-p]: 1883

User ID [-u]: dragino Password [-P]: Password

Certificate [--cert]: Certificate File Key [--key]: Key File

CA File [--cafile]: Certificate Authority File

Client ID [-i]: dragino-1d25dc

**Publish**

Enable Publish: ☒

Quality of Service [-q]: 0

Topic Format [-t]: CLIENTID/CHANNEL/data

Data Format [-m]: META

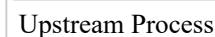
Must use META here

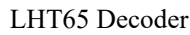
Publish must use META as data format for LG308

Step 4: Map the Device Address to Remote ID in MQTT server.



Step 5: Upstream: Save the change, we can see the log info via "sytem log", End Node and MQTT Server





### Subscribe

Enable Subscribe☒

Quality of Service [-q]

Topic Format [-t]

Save&Apply

Cancel

Subscribe to a topic

**Downstream Flow**

The image illustrates the downstream flow of data in an MQTT system. It consists of three terminal windows and several annotations:

- Top Left Window (Command Prompt):** Shows the execution of the command `mosquitto_pub.exe -h 10.130.2.192 -t t.dragino-1625dc -m "260211"`. An annotation states: "Use mqtt command to publish some info to the topic which LG308 is subscribing".
- Top Right Window (Serial COM9):** Displays the data received from the device (LG308). It shows a message "GPS NO FIX" and other telemetry data. An annotation points to this window: "Previous Log:".
- Bottom Window (mosquitto.exe Log):** Shows the MQTT broker's log. It records the connection of the client 't.dragino-1625dc' and the receipt of the message. An annotation points to the log entry: "MQTT Broker send this info to LG308".

The flow of data is as follows: The user publishes a message to the MQTT broker using the `mosquitto_pub.exe` command. The MQTT broker then sends this message to the device (LG308) over the serial connection. The device responds with its own data (e.g., "GPS NO FIX") back to the MQTT broker, which is visible in the log.

17/18

# Example For Different MQTT Servers

 <p><b>ThingSpeak Server</b> (<a href="http://www.thingspeak.com/">http://www.thingspeak.com/</a>)</p> <p>Examples</p>	 <p><b>乐联网平台</b> (<a href="https://www.lewei50.com/">https://www.lewei50.com/</a>)</p> <p>lewei Example</p>	 <p><b>AWS-IOT</b> (<a href="https://aws.amazon.com/cn/iot-platform/how-it-works/">https://aws.amazon.com/cn/iot-platform/how-it-works/</a>)</p> <p>AWS Examples</p>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

===End===

Retrieved from "[http://wiki.dragino.com/index.php?title=MQTT\\_Forward\\_Instruction&oldid=6997](http://wiki.dragino.com/index.php?title=MQTT_Forward_Instruction&oldid=6997)"

- This page was last modified on 21 August 2020, at 09:00.