# **Establish an MQTT framework architecture**

Raspberry Pi (Sensor subscriber)

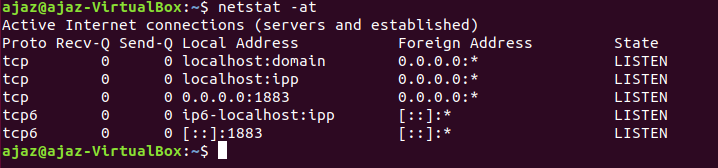
subscribes to pitch and roll

Raspberry Pi (CPU subscriber) subscribes to temperature and time

Raspberry Pi (Publisher) Publishes CPU temperature, Time and Sensor data as Pitch and Roll.

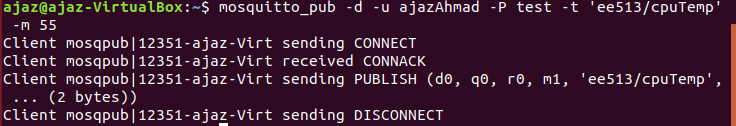
Virtual Box (MQTT Broker)

Below is the snapshot of running mqtt service on virtual box on port 1883:

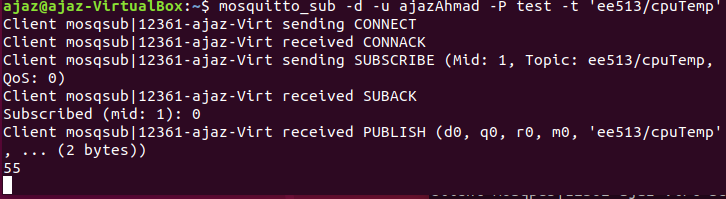


Testing if the framework works:

Below is the snapshot of publisher sending message on topic ‘ee513/cpuTemp’



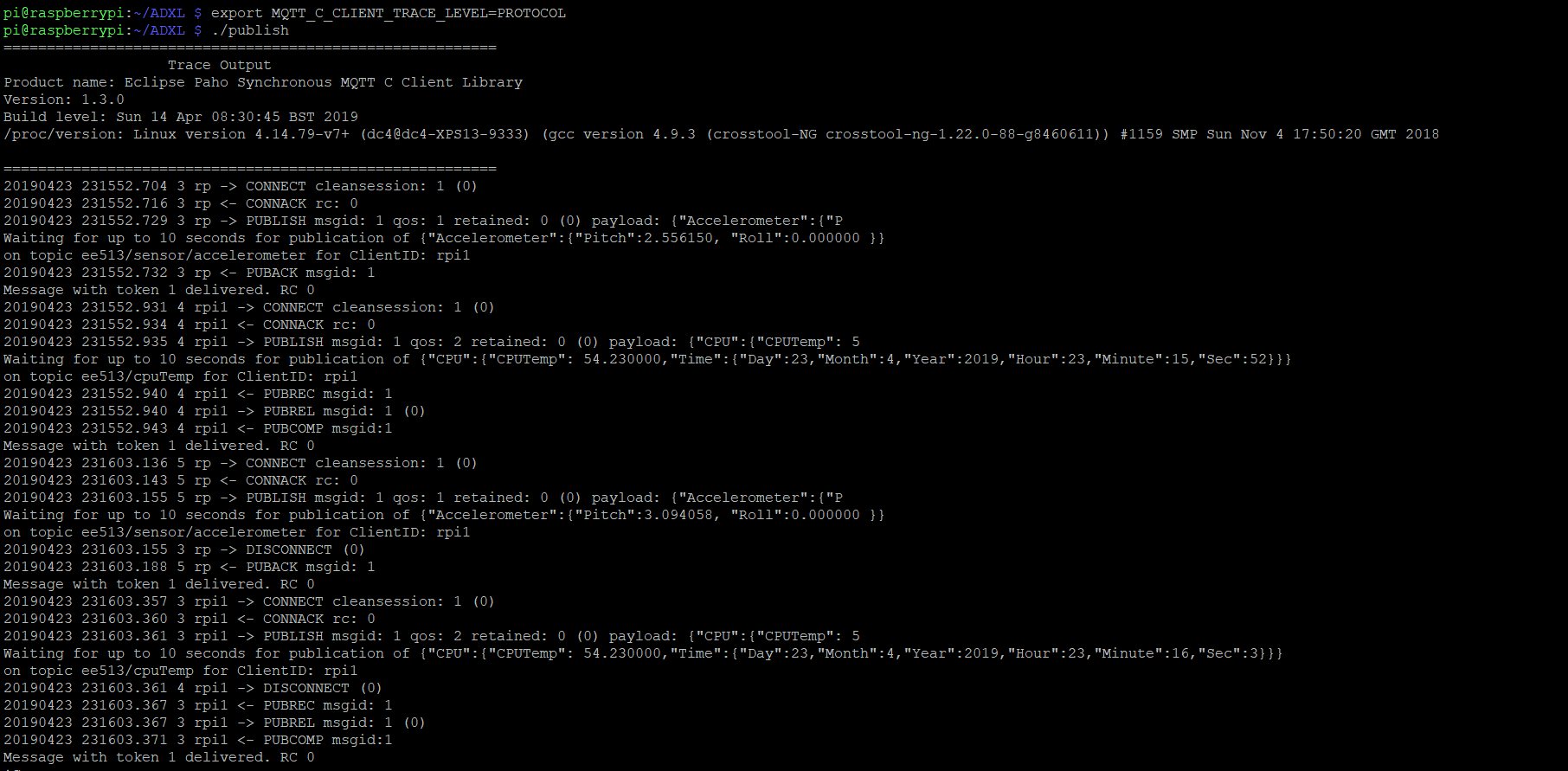
Subscriber receives the message and the screenshot is given below:



**Design and develop an MQTT publisher sensor application**

* The publisher is designed and defined in the file publish.cpp
* It publishes data under two different topics.
* The first topic is ‘ee513/cpuTemp’ and the second topic is ‘ee513/sensor/Accelerometer’.
* Under the first topic publisher publishes the current temperature of CPU and along with the current localtime in day/month/year HH:MM: SS format.
* Under the second topic it publishes pitch and roll from the sensor (accelerometer).
* The publisher publishes message in every ten seconds.
* The last will option is set for both the topics. On disconnecting the publisher abruptly the message received under topic ‘ee513/cpuTemp’ is “Lost Connection for CPU” and for ‘ee513/sensor/Accelerometer’ is “Lost Connection for Accelerometer”.
* g++ publish.cpp I2CDevice.cpp ADXL345.cpp -o publish -lpaho-mqtt3c.

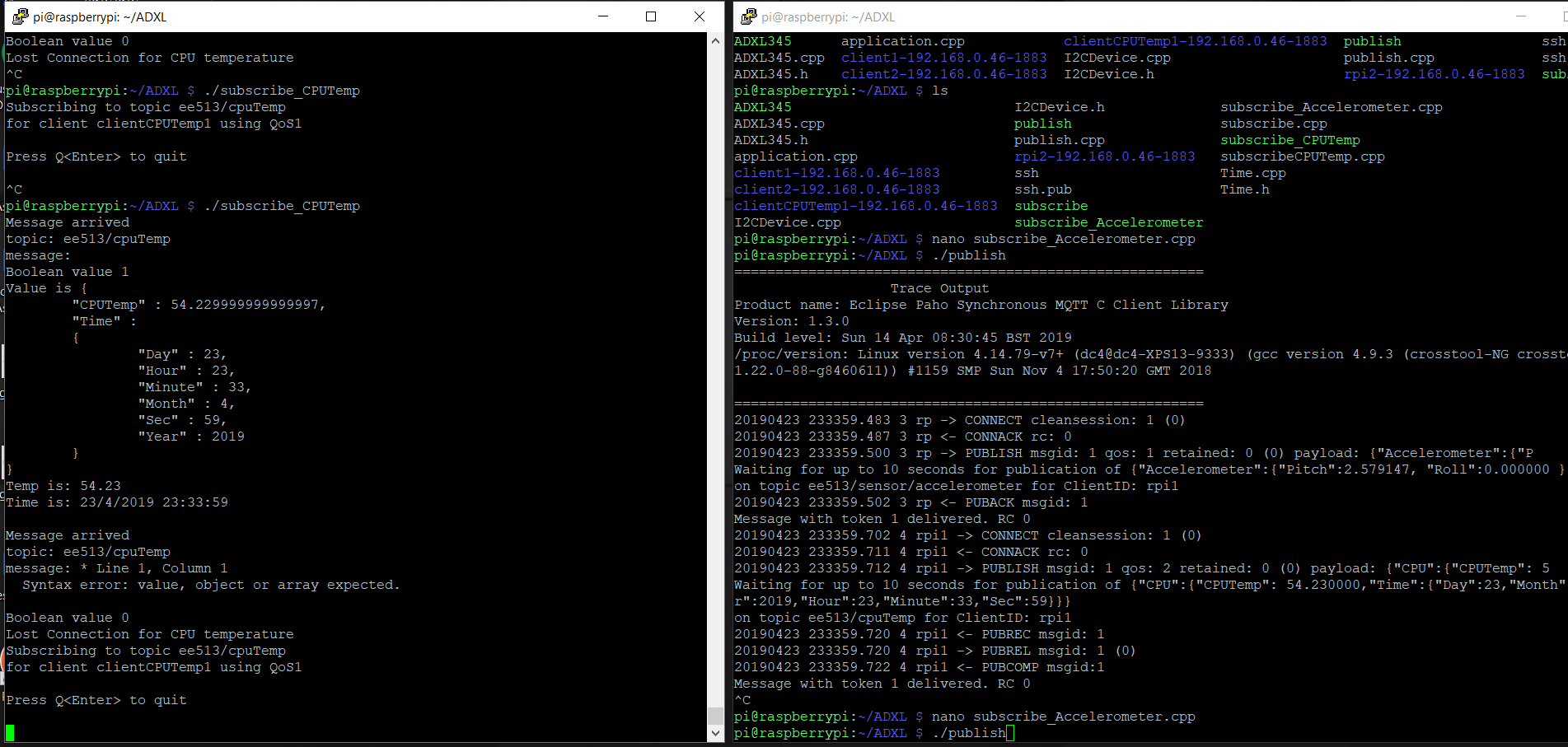
Trace details are given below:



The above screen displays tracing of publisher with different QoS level 1 and 2. In QoS with level 2 the publish received, publish release and publish complete can be observed in the sequence while with QoS 1 as the publish acknowledge is received the message is marked as delivered. The QoS level 2 is slow and has lot of overheads as compared to QoS 1.

**Design and develop multiple MQTT subscriber actuator applications**

* Two subscriber files are subscribeCPUTemp.cpp and subscribe\_Accelerometer.cpp.
* The led is turned on connected to GPIO4 when the CPU temperature goes above 55 and is turned down when temperature is below 55 for subscribeCPUTemp.
* The led blinks when the pitch value is positive for subscribe\_Accelerometer.
* The persistent connection is enabled for both the subscriber. For persistent connection, persistent is set to MQTTCLIENT\_PERSISTENCE\_DEFAULT and connection option cleansession is set to 0.
* To test the persistence, the subscribeCPUTemp is disconnected while the publisher continues publishing to broker. After, sometime subscribeCPUTemp is connected again and it receives the older messages of the publisher. Below is the snapshot:



* When the persistence was set to MQTTCLIENT\_PERSISTENCE\_NONE and cleansession was 1 then on reconnecting the client the new session was established and no previous message from the broker was received while the client was away.

**Design and develop a Qt MQTT visualization GUI application**

* The files associated to QT application are in the folder QTApplication.
* The Main Window has drop down item to subscribe to different topics.
* On activating any topic, the graph plot Y-axis is changed accordingly.
* Two methods are implemented to parse Json data, namely parseJsonData and parseSensorData as per the received topic respective methods are called.

