Lab Instructions:

Contents:

1. Publish and subscribe real time sensor data using python through MQTT protocol
2. Dump and analyze a short MQTT scenario using Wireshark Software
3. Explore the enhanced features of MQTT and analyze it in Wireshark
4. Display the sensor data using Node-red as subscriber
5. Arduino Controlled Robot arm using MQTT Protocol actuated by Gyro sensor.

PREPARATION QUESTIONS (TO BE ANSWERED BEFORE DOING THE LAB!)

1. The MQTT protocol is based on TCP/IP and both client and broker need to have a TCP/IP stack.
2. Both Publisher and Subscriber are considered as MQTT Client.
3. The MQTT connection is always established between a Client and broker, no client is connected to another client directly.
4. Once the connection is established, the broker will keep it open as long as the client doesn't send a disconnect command or it loses the connection.
5. The client identifier (short ClientId) is an identifier of each MQTT client connecting to a MQTT broker.
6. The Keep Alive is a time interval, the clients commit to by sending regular PING Request messages to the broker. The broker responds back with PING Response and this mechanism will allow both side to find if the other one is still alive and reachable.
7. The connection is initiated through a client sending a "Connect Command" message to the broker. The broker responds back with a "Connect Ack" and a status code. In the following table you can see all return codes at a glance.
8. what are the things that should be same in both the PUB_SUB scripts and why?
   Topic. QOS, username_pwd, Lastwill, Birth, Keepalive values. Since it is PUB_SUB based protocol, we need those to be same, otherwise the published data won't be received by Subscriber. But client ID should be unique for each clients.
9. How many QOS Levels are there in MQTT & what are the main differences?
   There are 3 QOS Levels.
   QOS=0: Fire & forget, No Acknowledgement
   QOS=1: Publisher deletes the stored message only after receiving 1 Acknowledgement from Broker
   QOS=2 Publisher deletes the stored message only after receiving 1 Acknowledgement from Broker & Broker deletes the stored message only after receiving Acknowledgement from Subscriber
10. What are the Special Features available in MQTT for IoT?
    Last will- If publisher is disconnected, Broker sends Publisher's Last will to subscribers in 60 seconds
    Keepalive- Clients pings the Broker for particular time interval to notify they are alive to receives messages
    Birth- Broker records when the publisher is born & started sending messages
11. Why is Quality of Service important?
    QoS is a major feature of MQTT, it makes communication in unreliable networks a lot easier because the protocol handles retransmission and guarantees the delivery of the message

13. What are the return codes and what are their uses?

Return Code	Return Code Response
0	Connection Accepted
1	Connection Refused, unacceptable protocol version
2	Connection Refused, identifier rejected
3	Connection Refused, Server unavailable
4	Connection Refused, bad user name or password
5	Connection Refused, not authorized

14. How will you find the IP address of the Raspberry pi and how many possible ways are there? (get familiar with Linux commands & Rpi)

    "Ifconfig" linux command tells the IP address of the Rpi

15. What do you need to access your IoT device regardless of distance, say like across the globe?

    In order to stay in the localhost network (IoT device) use VPN

1. How to Run Python code for BME280:

After setting up the BME280 Sensor on RPi

Please make sure you already have the following packages in RPi

1. MQTT Broker (Mosquitto)
2. Adafruit GPIO library
3. Paho MQTT Library

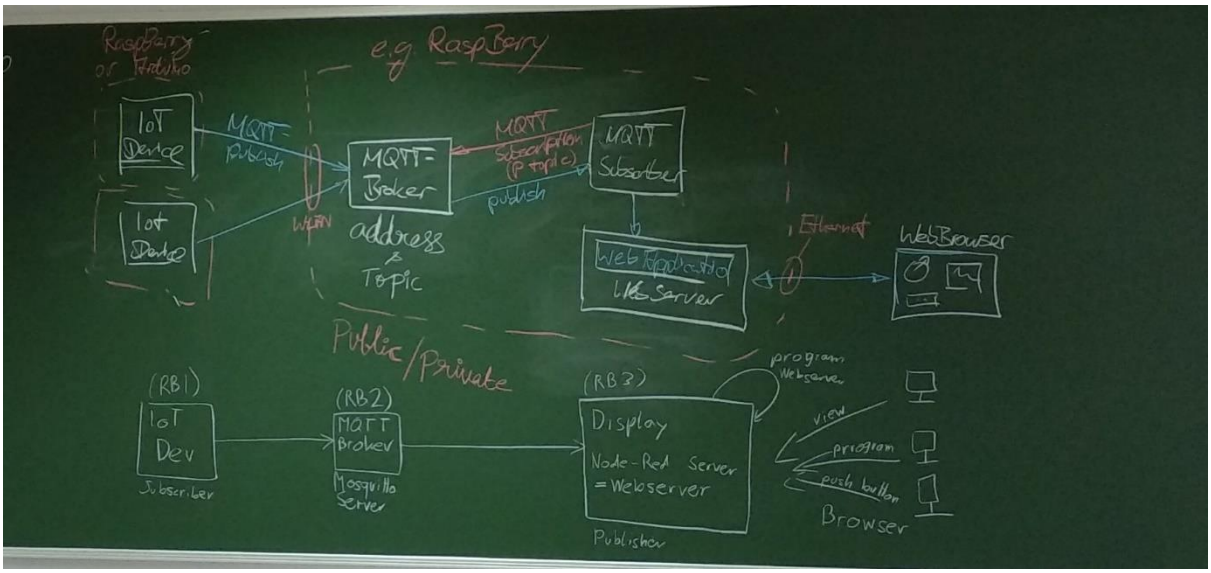

Figure 1: MQTT Setup

Access RPi in Putty:

- Open the Putty (SSH), Type your "username" and "password" with "IP address of **Client RPi**", Once you're done you will get access to Rpi as one of the multiple users. Make sure Your PC & both Rpi are connected in same network (fhintern for instance).
- Before executing the Python scripts for Publishing and subscribing, please check in both scripts if the following parameters match: Topics, IP address, QOS level, Broker's Username & Password

Note: Run the publisher first and then the subscriber

Now, Open the Linux terminal and run these Publishing and Subscribing Scripts to send & receive Real Time Sensor Datas via MQTT Broker.

1) The Python scripts are saved in directory**: /home/pi/Desktop/PUB_SUB**

>cd /home/pi/Desktop/PUB_SUB
>python publisher_easy.py
>python subscriber_easy.py

After playing around the sensor and observing its intentional variations (Touching the sensor increases pressure) stop° the Programs & go to Wireshark Analysis part.

ERROR DETECTION MECHANISM IN MQTT PYTHON PROGRAM:

1. To detect authentication error, we need to define the on_connect() method in MQTT PUB_SUB program.
2. The usage format is “on_connect(client, userdata, flags, rc):”
3. The rc is the return code.
4. The on_connect method prints out the return code. Looks like
   def on_connect(client, userdata, flags, rc)
   print("Connected with result code " + str(rc))

Reference:

- https://github.com/adafruit/Adafruit_Python_GPIO
- https://pypi.org/project/paho-MQTT/

2. How to dump MQTT packets & Analyze in Wireshark:

In order to analyze network scenario, we need to do the following:

- We are going to record the data packets in the Broker Rpi, so Switch to **Broker Rpi** in Putty by typing “Broker’s IP Address” in Hostname space of Putty and click “open”.

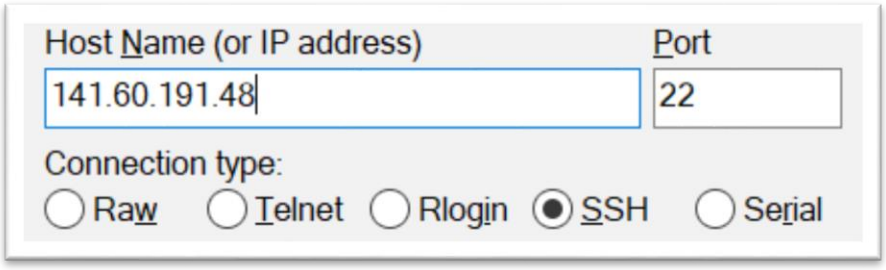

Figure 2: Putty window

- Once you are in, you will be asked for “Username & Password”.

  Username: pi

  Password: Masterproj

  (Only one common user in Broker*, so the students will observe everything in & out data packets in Broker)

- Same as in Terminal, type following Linux commands to record and analyze MQTT packets,

1) (*)To create a file in such a directory:
>sudo nano /home/pi/Desktop/capture.cap

2) Close the file capture.cap

>Ctrl+X

3) Make the empty file executable

> sudo chmod o=rw /home/pi/Desktop/capture.cap

#This allows other users (eg. Root User) to read/write this file (capture.cap)

4) Capture the traffic in Broker RPi using the light weight wireshark tool called “tshark”

To capture traffic from Putty: (Run tshark using below command)

>sudo tshark -c 500 -w /home/pi/Desktop/capture.cap

***Note**: The PUB_SUB program has to be run after started capturing the sensor Data in order to see the Connection establishment frames. So you may need two Putty windows, one for Client & other for Broker.*

5) After started capturing when the count down starts, now start° the PUB_SUB programs in Client Rpi. This is important to observe from Connection Establishment frame to the end. For this you have to open new putty window in parallel, type in the IP address of Client Rpi & then your group login credentials (Client Rpi has 8 users*)

Username: groupX

Password: GroupX

6) Create a folder named “Capture” in your C Drive to store the Captured file pulled from Broker to our PC.

7) To transfer the file from Broker RPI to our PC using command prompt: (FTP)

>pscp pi@141.60.191.41:/home/pi/Desktop/capture.cap C:\capture

(Command Usage format:pscp pi@IP:Fromdirectorypath Todirectorypath (Only three spaces in entire command)

8) Go to **C:\Capture** and Open the transferred (.cap) file with Wireshark (Make sure that you have the latest version of Wireshark in your pc to open MQTT packets)

9) Since we are using MQTT Protocol, filter the MQTT by typing “MQTT” in the filter bar on top blank space

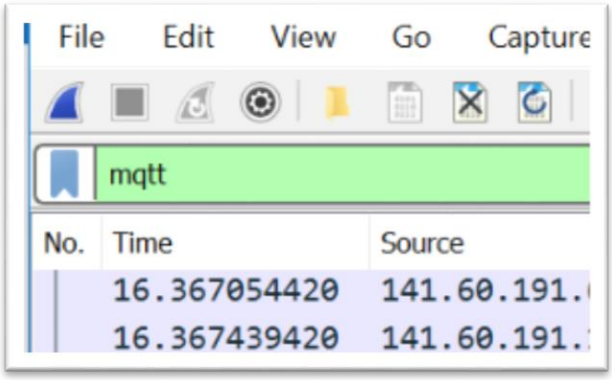

Figure 2.1: Filteration of MQTT

10) Observe the source and destination IP address of each frames along with their ports to know where all the sensor data travels

Time	Source	Source Port	Destination	Protocol	Destination Port	Info
16.367054420	141.60.191.67	35538	141.60.191.108	MQTT	1883	Connect Command
16.367439420	141.60.191.108	1883	141.60.191.67	MQTT	35538	Connect Ack

Figure 2.2: Wireshark Tabs

11) Also, in the last line of middle window, you will see MQTT protocol, where you can see the MQTT message contents of respective Traffic frames. Eg. Topics, Username_Password, Sensor data etc.

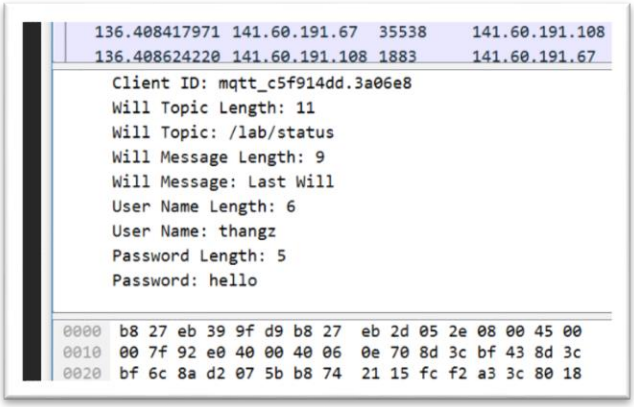

Figure 2.3 : MQTT features visualized in Wireshark

Reference:

- https://www.ssh.com/ssh/putty/windows/
- https://stackoverflow.com/questions/42278560/file-transfer-on-raspberry-pi-using-pscp
- https://www.ssh.com/ssh/putty/putty-manuals/0.68/Chapter5.html

3. EXPLORE THE ENHANCED FEATURES OF MQTT AND ANALYZE IT IN WIRESHARK:

Above we have seen the Normal MQTT Packets, Now we will investigate the enhanced features of MQTT protocol.

3.1. HOW TO CONFIGURE MQTT BROKER TO REQUIRE CLIENT AUTHENTICATION USING USERNAME AND PASSWORD (PYTHON):

1. We need to copy the password file into the etc\Mosquitto folder and then edit the mosquitto.conf file to use it. To open the mosquitto.conf file,

>sudo nano /etc/mosquitto/mosquitto.conf

2. In the mosquitto.conf, set allow anonymous to false and to set the password_file path.

3. To do that copy and paste the following two lines in the file:

allow_anonymous false
password_file/etc/mosquitto/passwd

4. Save & Exit the passwd file

>ctrl+x

Now try to run the PUB_SUB programs:

>cd /home/pi/Desktop/PUB_SUB
>python publisher_easy.py (Connection without authentication)
>python subscriber_easy.py(Connection without authentication)

Connection Failed! Then think why it is failed! Do the needfull!

Hint: Now we Authenticated the broker. Check the meaning of error message rc.

5. Use the method username_pw_set() of the Paho client.
6. The usage format is

>client.username_pw_set(username="uname",password="passwd")

7. Insert this line in Publisher program to call this method.
8. Run the enhanced version with user access authentication command.

>python publisher_enhanced.py (Connected to broker)
>python subscriber_enhanced.py (Connected to broker)

Task: After configuring the broker, observe the MQTT scenario using wireshark.
Inference: what difference did you find now? Please discuss with professor.

References: http://www.steves-internet-guide.com/MQTT-username-password-example/
https://www.hivemq.com/blog/MQTT-security-fundamentals-authentication-username-password

3.2 QoS LEVELS IN MQTT

HOW TO REALIZE DIFFERENT LEVELS OF QOS IN MQTT USING PYTHON:

We need to use client.publish method to publish a message on a given topic and assign a Quality of Service to be used for the message. Change the QOS level in the higlighted area shown below in PUB_SUB program.

The usage format is client.publish(topic,payload,qos,retain)

client.publish("/lab/hum", humidity, 1, 1)

client.publish("/lab/temp", degrees, #qos 1, 1)

client.publish("/lab/press", hectopascals, 1, 1)

Parameters: mid — the function will set this to the message id of this particular message.

payload (String) — Message payload to send.

QoS — Quality of Service to be used for the message.

retain (true, false) — set to true to make the message retained.

References:
https://dzone.com/articles/internet-things-MQTT-quality
https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.dev.doc/q029090_.htm

3.3. WIRESHARK ANALYSIS TASKS:

Now we explored the enhanced features of MQTT, let us capture traffic for different cases using general procedure of Wireshark capturing (*) (Tshark in Rpi). For instance modify somethings in our Publisher & Subscriber as follows:

- Start Capturing the data packets and then run the PUB_SUB program, otherwise we won't see connection establishment frame. (username, pwd)

For all tasks you need to start from (*) of the Wireshark Analysis part above,

1. With User authentication (Enter **client.username_pw_set("usrname","passwd")**)
2. Without authentication (Remove **client.username_pw_set("usrname","passwd")**)
3. With different QOS levels (**client.publish ("/lab/temp", degrees, #qos 1, 1**)
4. With Lastwill, Birth, keepalive features in both publisher and subscriber programs
5. Also, Without those features
6. Reduce/ Increase the parameters (temp, hum, press)in python subscriber

Note:

- Then Start Capturing the data packets and then run the pub sub program, otherwise we won't see connection establishment frame. (username, pwd)
- For better understanding use **500-600 capture frames** in all cases, only for lastwill use **3000 frames** since the message time to reach the subscriber takes a while (60 sec)
- To test Lastwill feature, make sure that you will disconnect the Ethernet Cable / USB cable of Publisher, so that it's Lastwill reaches the Subscriber (**Abnormal Disconnection**)
- Also, you can stop the Publisher python script in the Linux Terminal (**Normal Disconnection**)
- Lastwill message takes 60 seconds to reach the Subscriber
- Basically MQTT works with TCP Protocol

Reference:

- https://www.wireshark.org/docs/dfref/m/MQTT.html
- https://www.wireshark.org/
- https://www.snowfactory.com/webcam/uncategorized/raspberry-pi-install-tshark-sniffing-tool/
- http://MQTT.org/
- https://pypi.org/project/paho-MQTT/
- http://www.steves-internet-guide.com/publishing-messages-MQTT-client/

4. How to Create Node-RED flows and Visualize Sensor Data in Dashboard:

So far we have seen python as Publisher & subscriber, but now we are going to replace subscriber as Node-red flow.

Usually, the Raspbian has Node-red by default, we just have to install the Node-red Dashboard node-modules from the web workspace.

TO START THE NODE RED WORKSPACE SESSION:

>sudo node-red

TO OPEN THE WEB WORKSPACE FOR NODE-RED OPEN THE BROWSER & TYPE IN ADDRESS BAR:

http://<IP address of your Pi>:188*(Your Port number)**

1) Run the Publisher Python program in the terminal for sending the data to MQTT broker!
>**python /home/pi/Desktop/PUB_SUB/publisher_enhanced.py**
(check the IP address of the broker before running the programs)

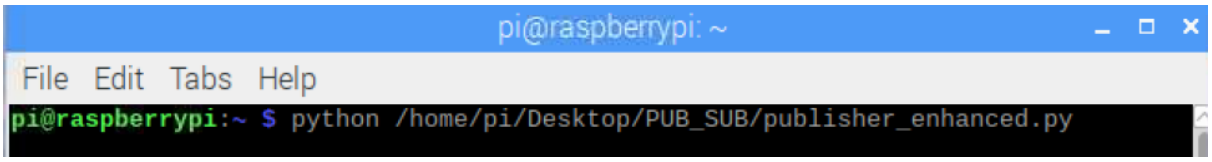

Figure 4.1: Terminal command for running publisher python program

2) Start the Subscriber Node-red in terminal
>node-red

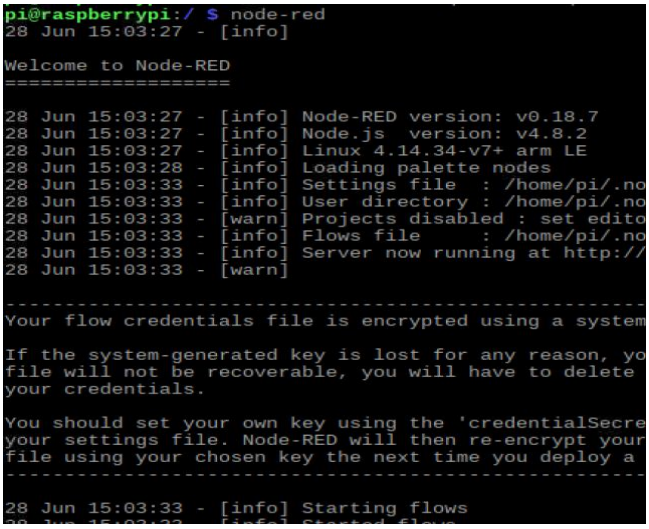

Figure 4.2: Terminal command for running Node-red session

3) Open the browser in your laptop and type the following **(**http://*IP ADDRESS OF BROKER RPI*:1880**)** then the general node-red Workspace will be seen as below:

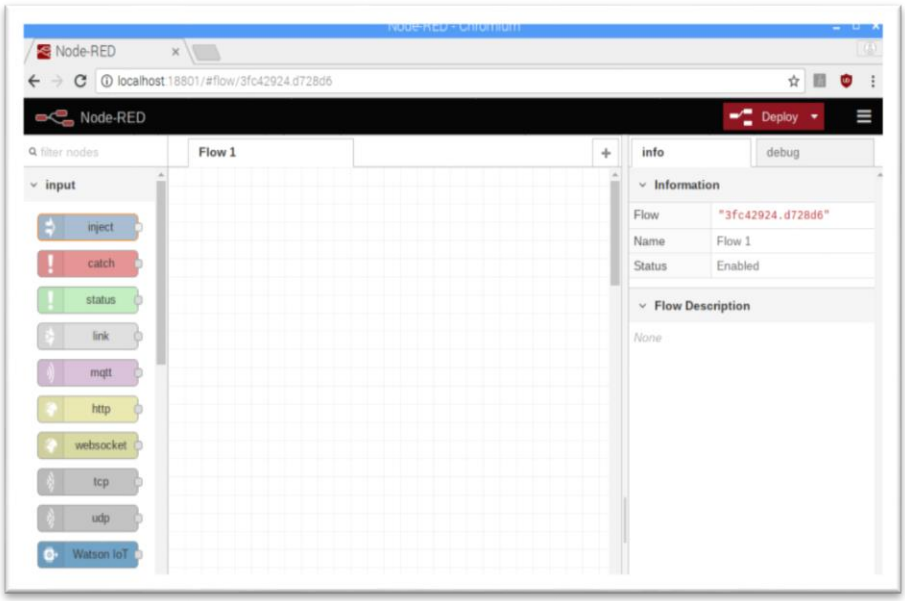

Figure 4.3: Node general workspace

4) Three nodes are required to display the sensor Data **BME280** which are shown below:

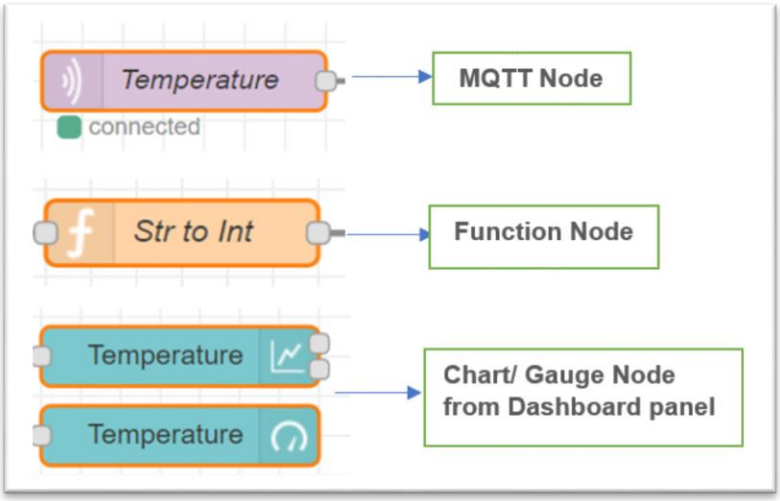

Figure 4.4: Nodes to be used

5) Type the four nodes in “Filter nodes tab” in top left corner of the node-red window and drag and drop everything to the workspace area then as shown in the figure below connect the nodes:

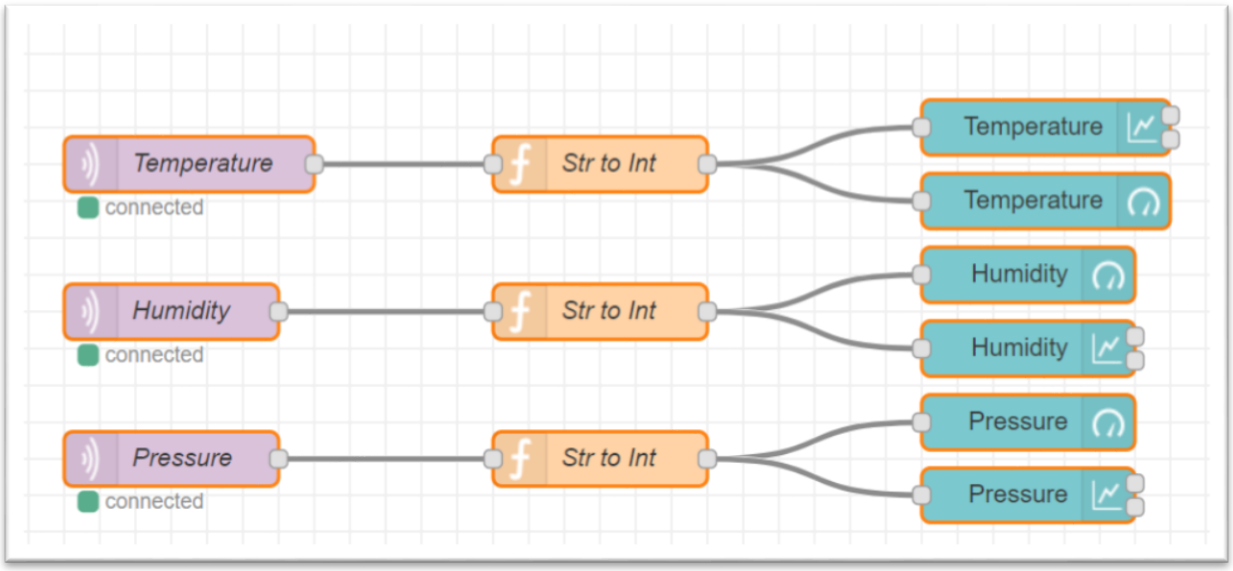

Figure 4.5: Flow Design for BME280

6) **Starting to configure all the nodes:**
Double Click MQTT node, provide the “Topic” of published Data (/lab/temp), mention “QOS Level”, Specify the node “Name” of your wish & Click on Pencil icon right next to the “server” tab

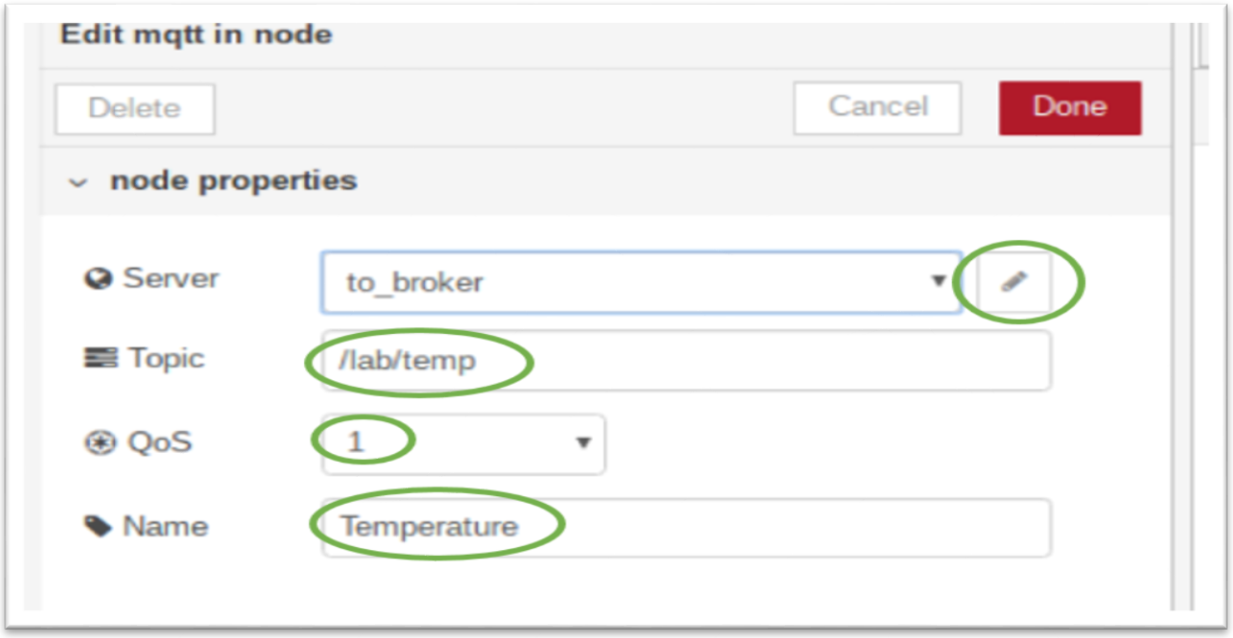

Figure 4.6: Flow Design for BME280_ Configuration

7) **In the server tab:**
Specify the “Name” of the server as your wish, specify the “Keepalive time”, give the IP address of your MQTT Broker along with Port number of our broker Specified (see below):

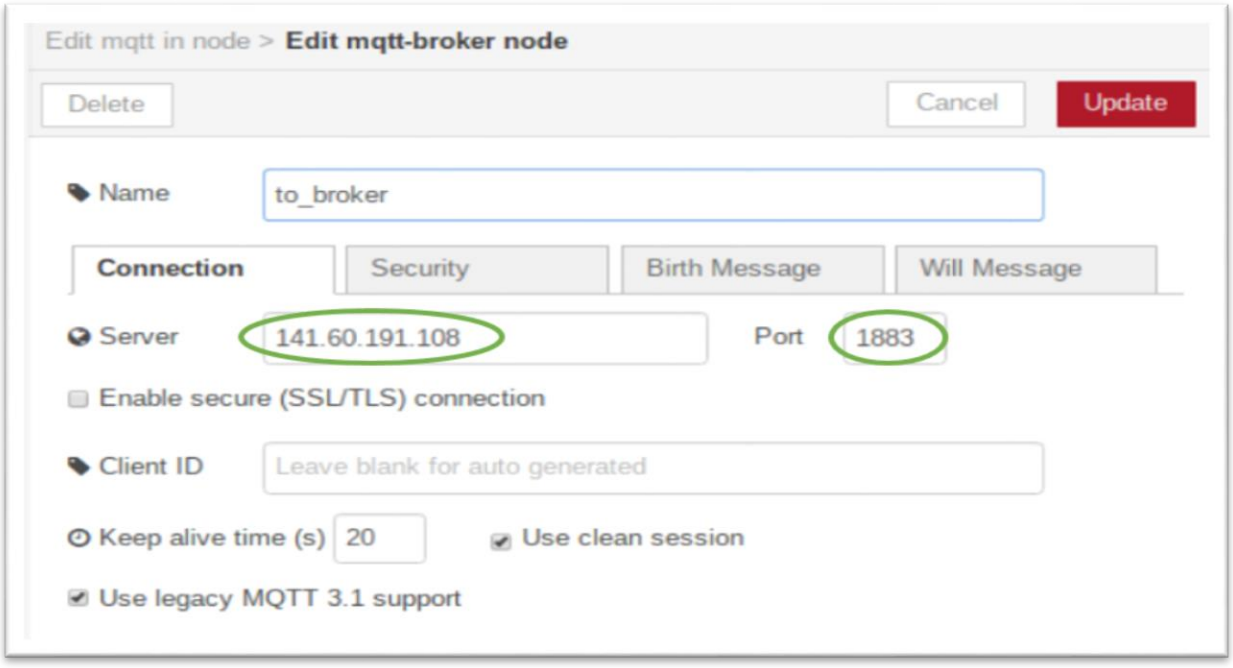

Figure 4.7: Connection Edit tab_ Configuration

8) **Go to security Tab:**
Specify the username and password of the Broker

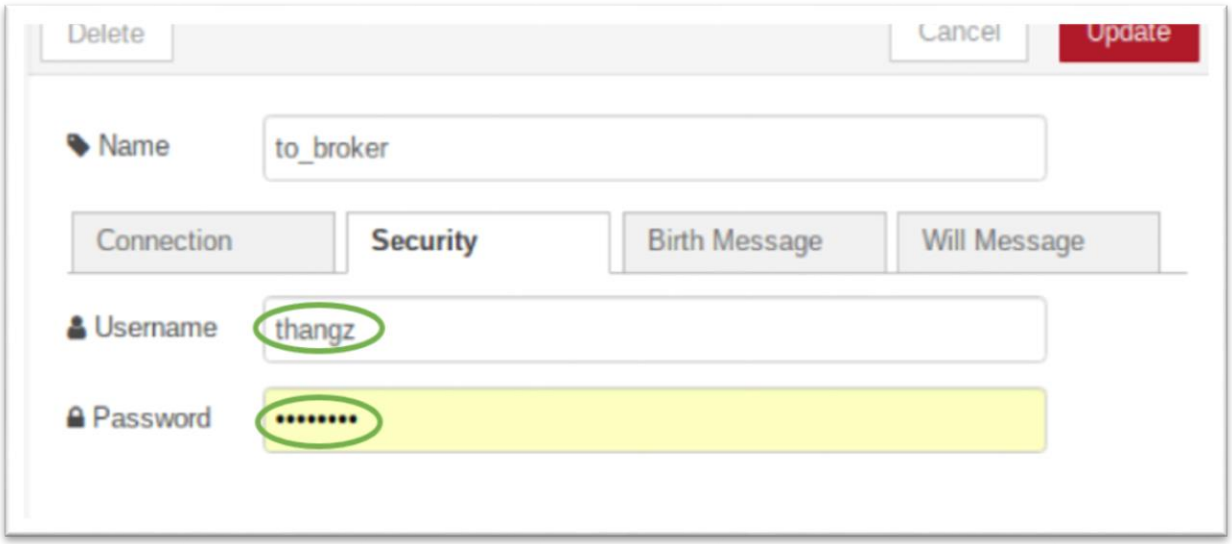

Figure 4.8 : Security Edit tab_ Authentication

9) **Go to Birth Message:**
Set “Retain” as **True** to receive retained messages, also give the payload as your wish (just as heading of our birth message)

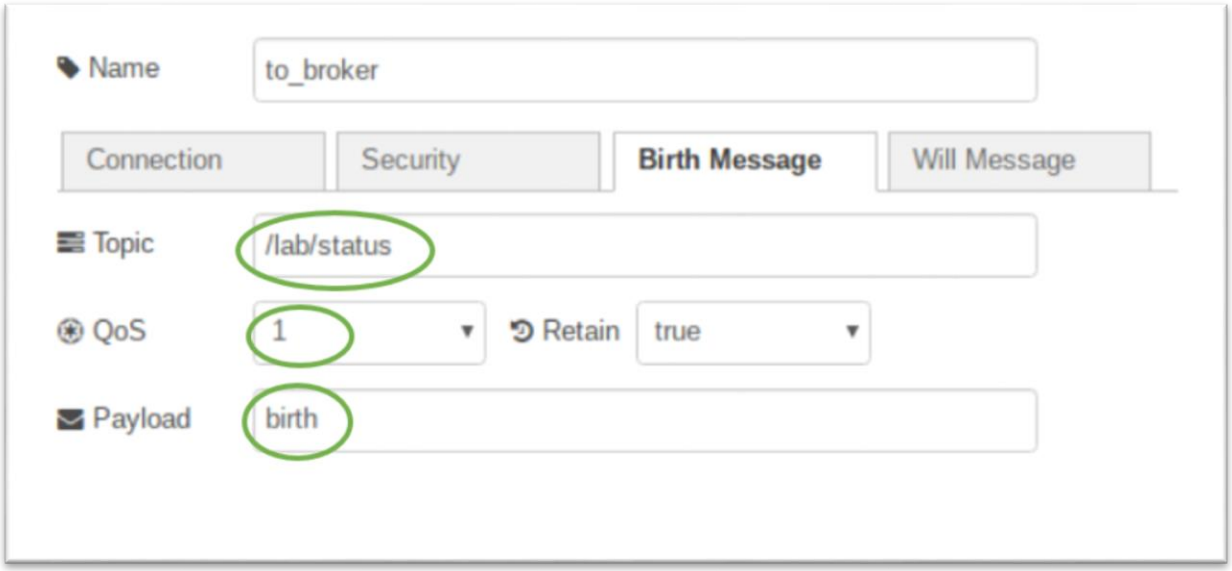

Figure 4.9: Birth Message Edit tab_ Configuration

10) **Go to will message:**
Set “Retain” as **True** to receive retained messages, also give the payload as your wish (just as Label of our Last Will message)

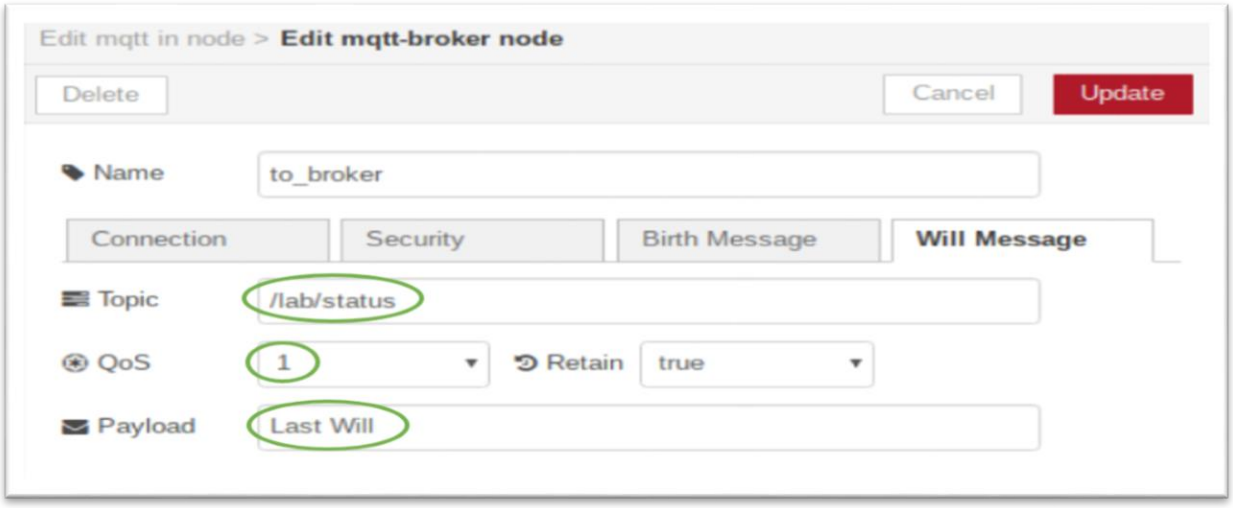

Figure 4.10: Will message Edit tab_ Configuration

After doing this, close the MQTT Node configuration.

11) **Go to Function node:**
Copy the Java script which runs inside the “function node” for converting string to integer (See below)

Figure 4.11: Function node edit tab_Program

12) **Go to Chart/Gauge:**
Mostly everything is already pre-configured, only we have to specify “Group Name” & “Labels”, if necessary specify the units and limits of units (min & max)
Main thing to remember is the groups with same name are aligned vertically therefore, we give Temperature [Gauge] even for the chart node as Group name to display it below the Gauge reading of temperature data in dashboard
Then check the configuration (Edit chart mode) of Chart/Gauge node by comparing the figures shown below:

Chart Configuration:

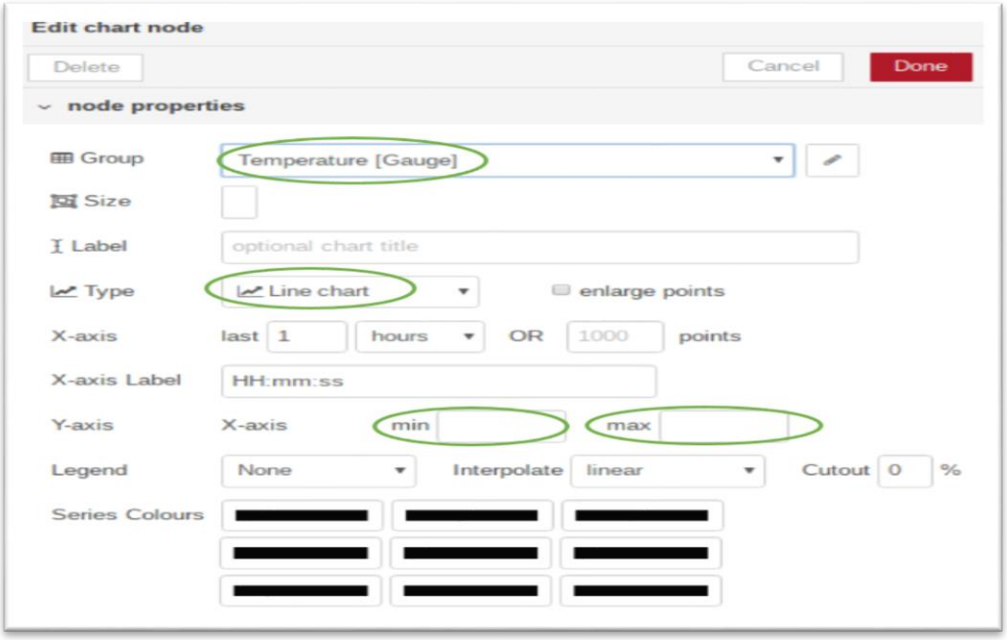

Figure 4.12: Chart Edit tab_ Configuration

Gauge Configuration:

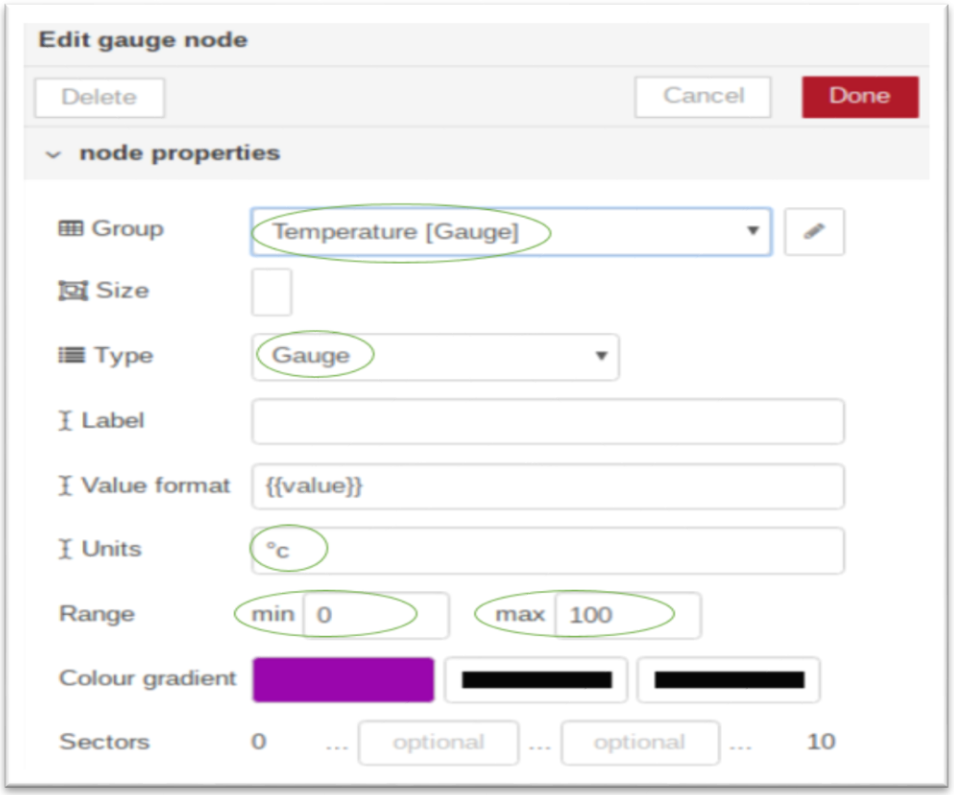

Figure 4.13: Gauge Edit tab_Configuration

After configuring all the nodes, deploy the flow by clicking “Deploy” (in top right corner of the window)

13) Open the node-red dashboard to visualize the sensor data by typing http:///*IP ADDRESS OF BROKER RPI*:1880/ui in the browser address bar

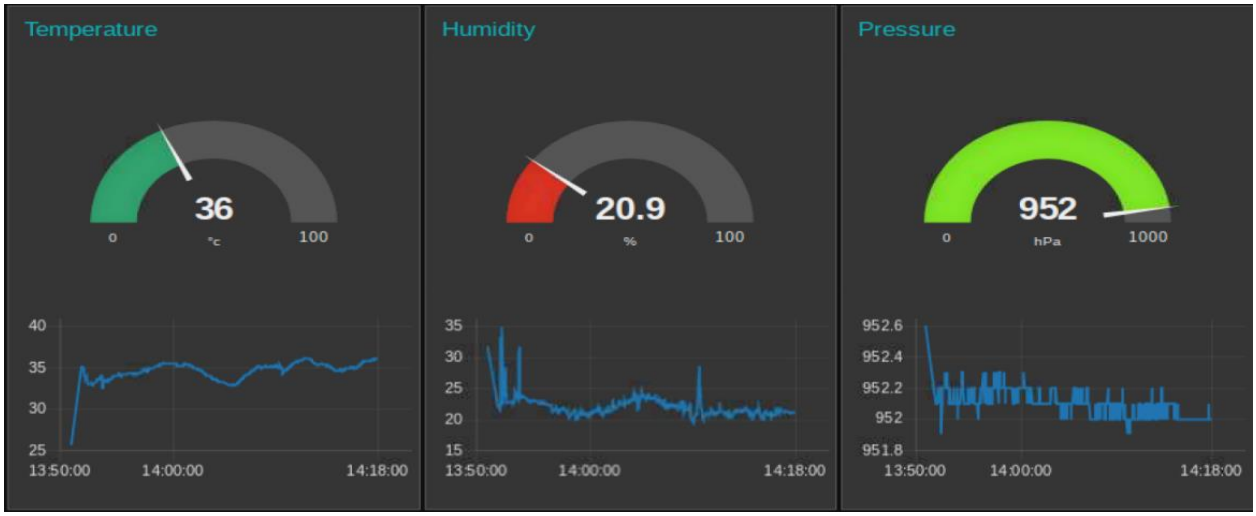

Figure 4.14: Dashboard View

Note: We have [Gauge] in Chart also in order to get CHART AND GAUGE in the same coloumn (like above one)

14) **Optional:** Now you could see the real time sensor datas coming from Raspberry A is received and visualized in Web GUI Dashboard. You can also view it as decoded messages if you connect a debug node to function node

Figure 4.15: Debug Node in under output nodes

Reference:

- https://oneguyoneblog.com/2017/06/20/mosquitto-MQTT-node-red-raspberry-pi/
- https://www.npmjs.com/package/node-red
- https://nodered.org/docs/hardware/raspberrypi
- https://flows.nodered.org/node/node-red-dashboard

5. ARDUINO CONTROLLED ROBOT ARM VIA MQTT PROTOCOL ACTUATED BY GYRO SENSOR:

1. Type the following nodes in “Filter nodes tab” in top left corner of the node-red window and drag and drop the node to the workspace area and connect the nodes then as shown in the figure below.

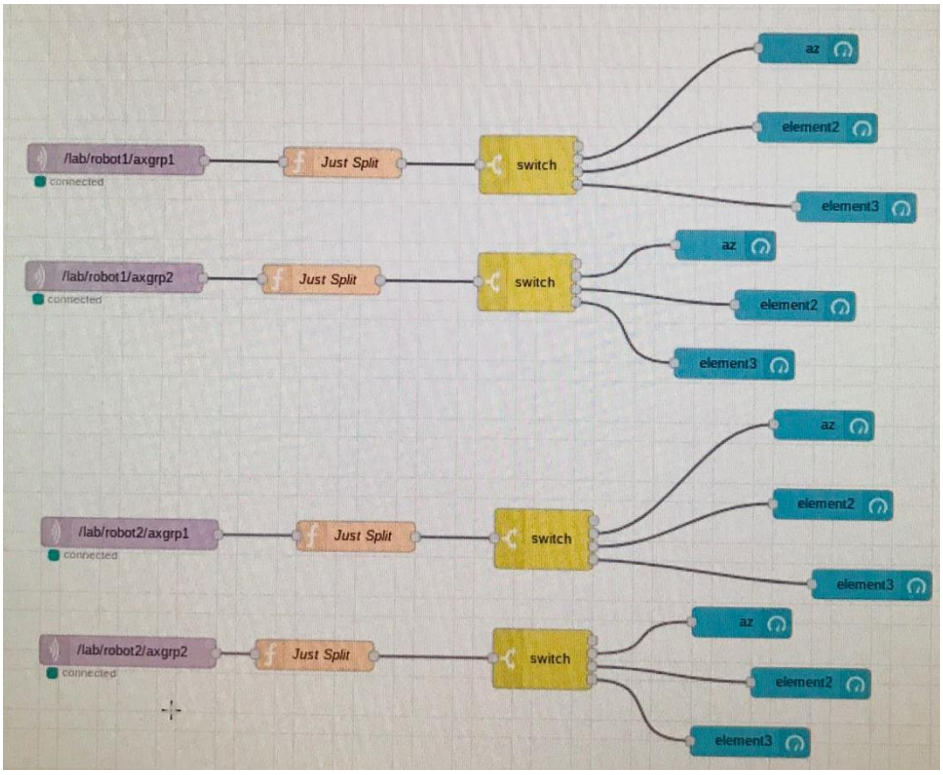

Figure 5.1: Node Flow for Robot control Visualization

2. Double Click MQTT node, provide the “Topic” of published Data (/lab/robot1/axgrp1), mention “QOS Level”, Specify the node “Name” of your wish & Click on Pencil icon right next to the “server” tab.

3. Click the function node and name it as “Just Split” and enter the following program in the function node and Click Done.

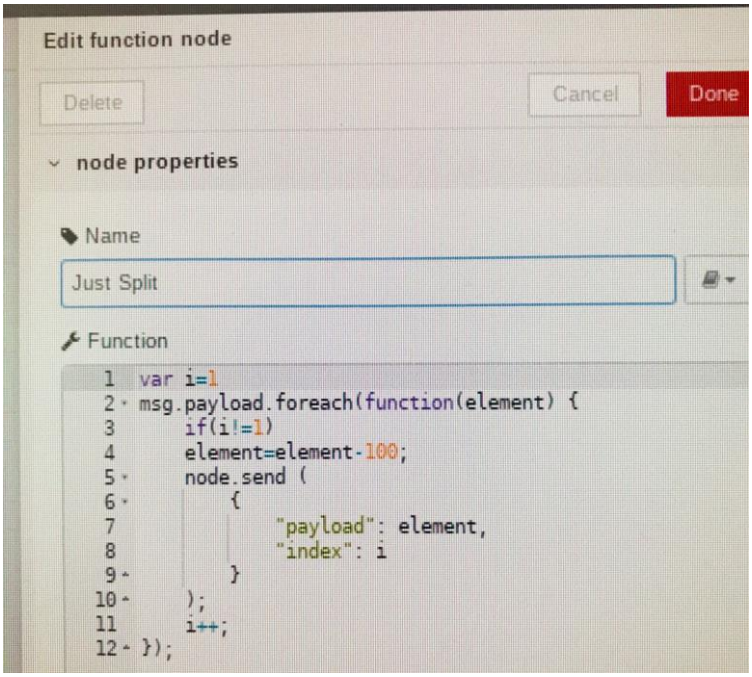

Figure 5.2: Function Code for splitting arrays

4. Click the Switch node and Click Add Button in the bottom of the property and add three values (X, Y, Z axes) and Select the index values as ‘1’ ‘2’ ‘3’ ‘4’.

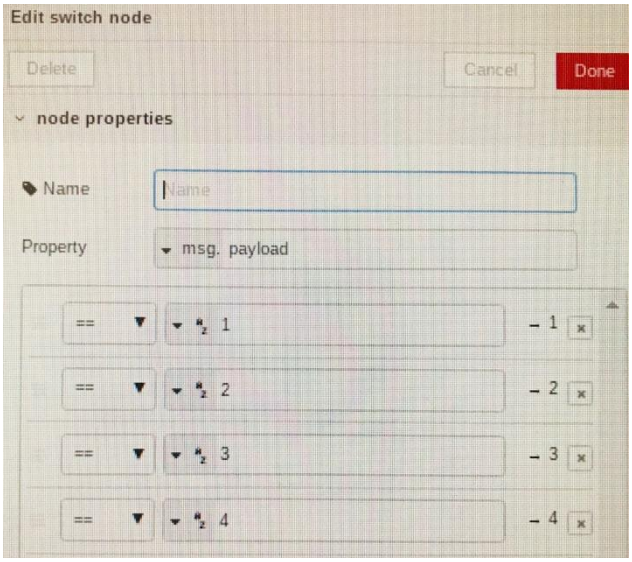

Figure 5.3: Switch Node configuration

5. In the Gauge node, Specify the details as given in the below image:

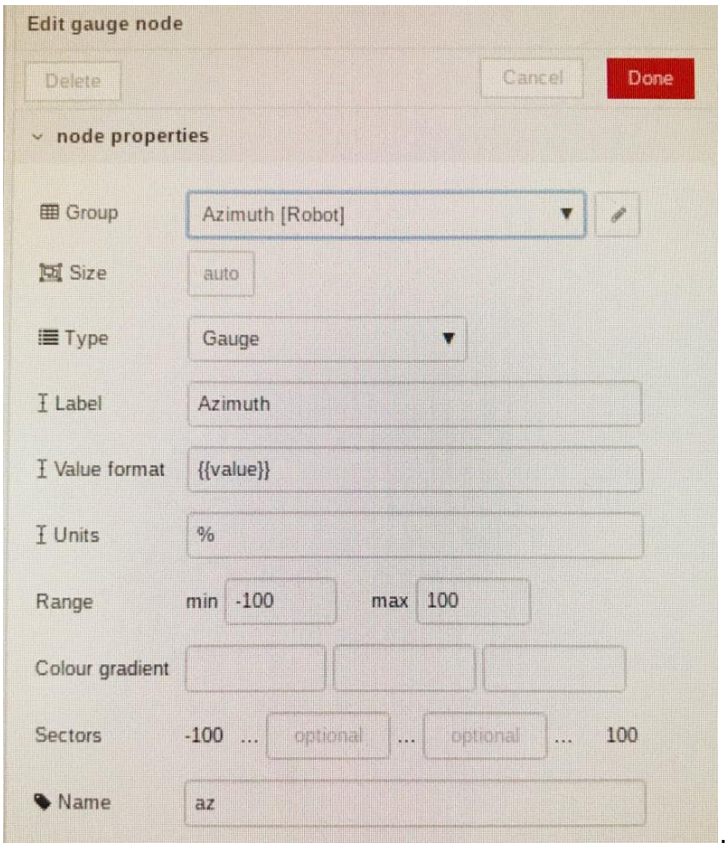

Figure 5.4 : Gauge Configuration for robot angle

6. Now, Deploy the project and Connect the Gyro with battery and tilt to simulate position of robot i.e. Publisher Gyro sends angle arrays to Broker and from broker, it reaches the Robot Subscriber.

Suggestions:

Let the students figure out some errors and correct it (related to MQTT)

The Client IDs of PUB_SUB are “Arduino Client”, Open both the Programs in desktop and let the students run and see what happens, both the clients connects and disconnects repeated because of infinite loop in both programs. To fix this issue, one of the Client ID should be changed. For instance, Client ID in Publisher Program should be changed from Arduino Client to Gyro.

Things to be changes: (Solution!)

- **In communication part of the Publisher Gyro program:**

```
// Attempt to connect
  if (MQTTclient.connect ("~~Arduino Client~~” “Gyro")) {  //Here is the the mistake that both the clients
                                                            PUB_SUB  had same client IDs and student should change one of them
  Serial.println("connected");        // So each time one connects to broker other gets disconnected
 // Once connected, publish an announcement // So the disconnected client runs reconnect loop
  MQTTclient.publish(topic,"connected");
```

Fig: Reconnecting Loop Error

Same kind of tasks could be given for python programming as well.

Inference: Each Clients should have unique client ID to avoid disconnections, but same other MQTT paramters like Topic, QOS etc.

Reference:

- https://nodered.org/docs/getting-started/first-flow
- https://groups.google.com/forum/#!topic/node-red/ThYtMXIZ81o
- https://flows.nodered.org/node/node-red-contrib-splitter
- https://nodered.org/docs/writing-functions
- https://pubsubclient.knolleary.net/api.html
- https://forum.arduino.cc/index.php?topic=483722.0