Hands-On Practice

Raspberry Pi as a MQTT Broker

SMARTBRIDGE
Let's Bridge the Gap

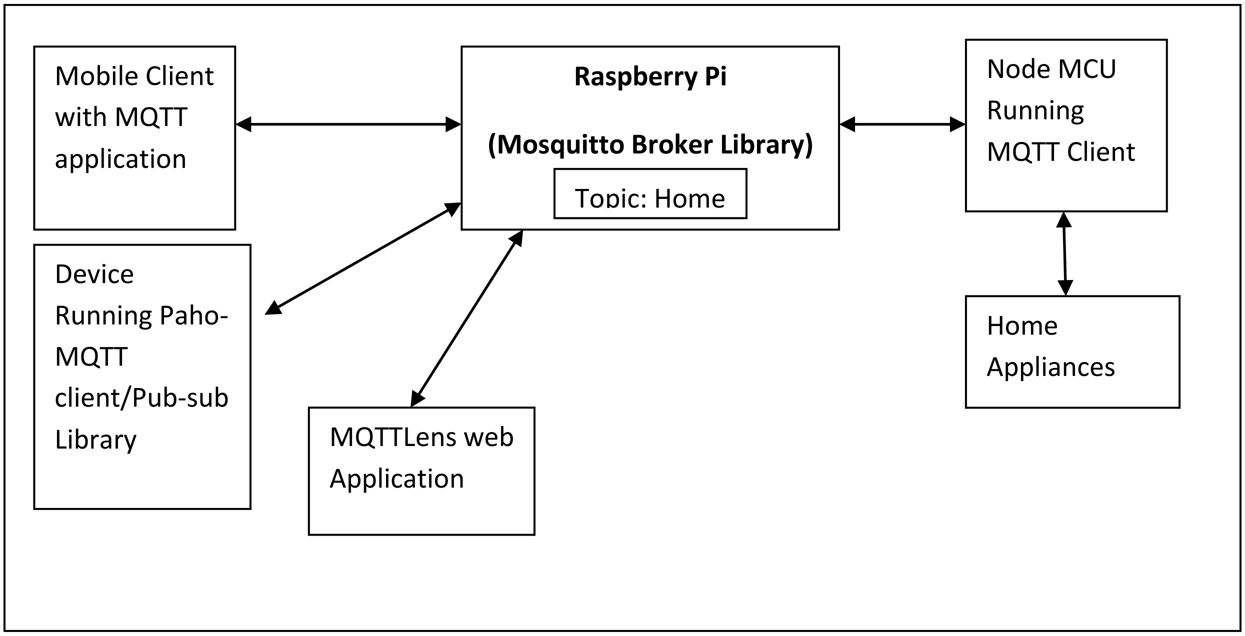
Raspberry Pi as a MQTT Broker

Introduction:

MQTT is the protocol of choice for M2M and IoT Applications. However, when it comes to selecting the MQTT broker, most of the times we resort to Cloud based Brokers. Having a local MQTT Broker may have many advantages over Cloud based Brokers, like Security, Flexibility, Reliability, Low Latency, Cost Effectiveness, better QoS implementation etc.

Experiment:

In this experiment raspberry pi acts as MQTT broker in the local area network using Mosquitto python library. Other devices in the network can connect to this broker through MQTT protocol on port number 1883. The devices can publish or subscribe to the topics to exchange the data.


Mobile Client with MQTT application
Raspberry Pi
(Mosquitto Broker Library)
Topic: Home
Node MCU Running MQTT Client
Device Running Paho-MQTT client/Pub-sub Library
Home Appliances
MQTTLens web Application

Mosquitto MQTT Broker:

Mosquitto is an open source iot.eclipse.org project. It implements the MQTT protocol versions 3.1 and 3.1.1. For more details please refer to http://mosquitto.org/

Mosquitto on Raspbery Pi:

Raspberry Pi has enough compute power to run Mosquitto and function as a personal MQTT Broker which can cater most of our personal MQTT needs. So let’s go ahead and explore Installation, Testing and Un installation process.

What do you need?

- Raspberry Pi with Raspbian Operating System
- Ethernet / WiFi Connection to Internet from Raspberry Pi
- NodeMCU Dev. Board with PubSub library
- MQTT Lens, Chrome web App
- *Note: All the devices or applications shall be on same network.*

Install Mosquitto MQTT Broker:

1.0 SSH into Raspberry Pi or Open Terminal and create a new directory for temp files –

```
mkdir mosquitto

cd mosquitto
```

2.0 Import the repository package signing key –

```
sudo wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key

sudo apt-key add mosquitto-repo.gpg.key
```

3.0 Make the repository available to apt –

```
cd /etc/apt/sources.list.d/

sudo wget http://repo.mosquitto.org/debian/mosquitto-jessie.list
```

4.0 Install Mosquitto MQTT Broker –

```
sudo apt-get install mosquitto
```

5.0 Check Mosquitto Service Status, Process and Default Port (1883) –

```
sudo service mosquitto status

sudo ps -ef | grep mosq

sudo netstat -tln | grep 1883
```

If you see Mosquitto service running and listening to TCP Port 1883, you have a functional MQTT Broker.

6.0 Test Mosquitto MQTT Broker with MQTT Client:

For testing you can use any MQTT Client. However, if you have Python 2.7 Installed on your machine, you can test it with following sample Python scripts. To execute these Scripts, you must have Paho MQTT Client installed on your machine. You can install it with pip command –

sudo pip install paho-mqtt

Once Paho Client Library is installed, you can download and execute following Python scripts (Don't forget to change "MQTT_BROKER" IP Address) –

7.0 Uninstall Mosquitto MQTT Broker:

To uninstall Mosquitto you can use following command –

sudo apt-get purge mosquitto

If you want to completely remove Mosquitto with it's associated configuration files, use following command –

sudo apt-get --purge remove mosquitto

MQTT publisher:

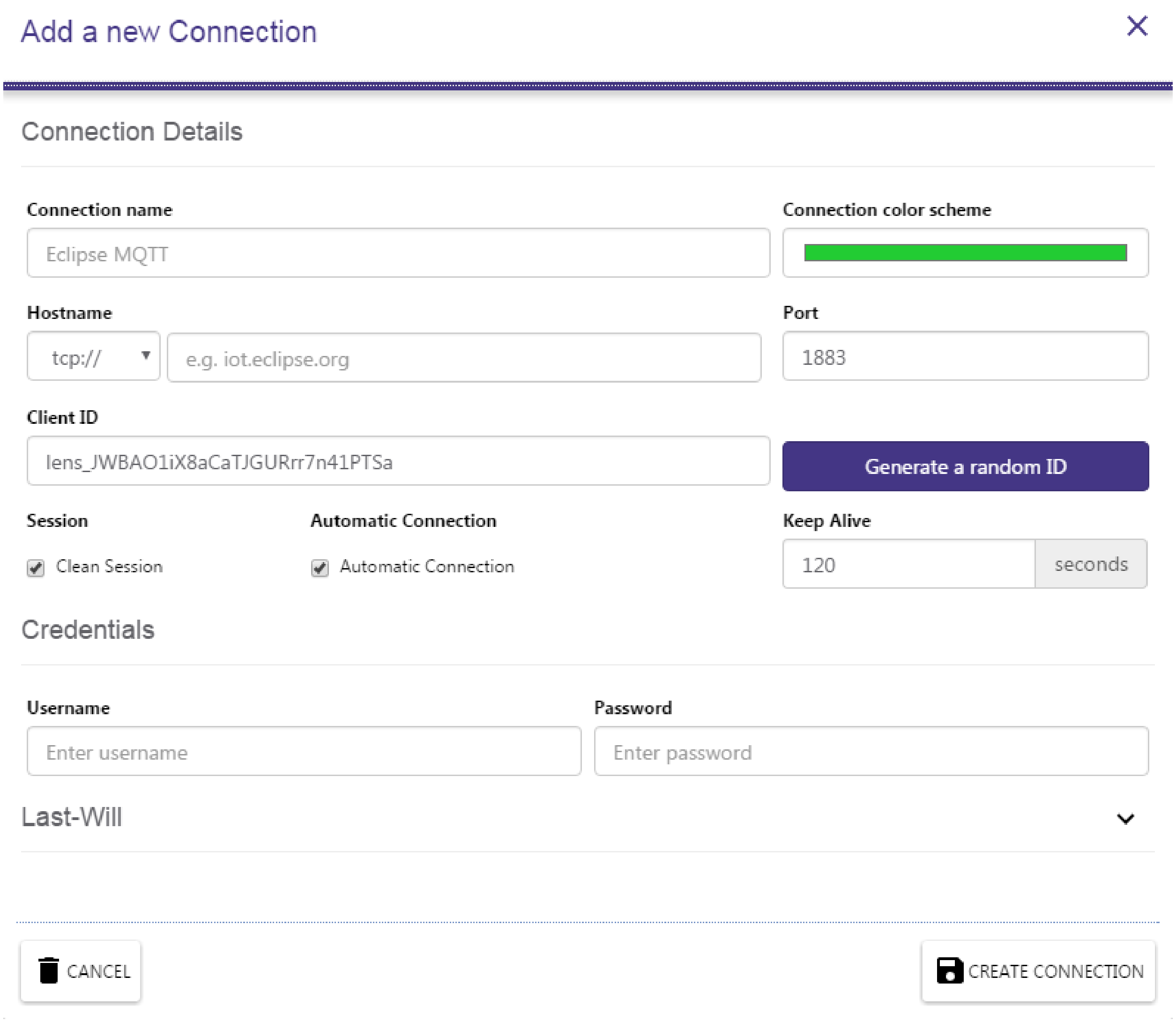
```
# Import package
import paho.mqtt.client as mqtt

# Define Variables
MQTT_BROKER = "MQTT Broker IP or DNS Name"
MQTT_PORT = 1883
MQTT_KEEPALIVE_INTERVAL = 45
MQTT_TOPIC = "testTopic"
MQTT_MSG = "Hello MQTT"


# Define on_connect event Handler
def on_connect(mosq, obj, rc):
        print "Connected to MQTT Broker"

# Define on_publish event Handler
def on_publish(client, userdata, mid):
        print "Message Published..."

# Initiate MQTT Client
mqttc = mqtt.Client()

# Register Event Handlers
mqttc.on_publish = on_publish
mqttc.on_connect = on_connect
```

```
# Connect with MQTT Broker
mqttc.connect(MQTT_BROKER, MQTT_PORT, MQTT_KEEPALIVE_INTERVAL)

# Publish message to MQTT Topic
mqttc.publish(MQTT_TOPIC,MQTT_MSG)

# Disconnect from MQTT_Broker
mqttc.disconnect()
```

MQTT subscriber:

```
import paho.mqtt.client as mqtt

# Define Variables
MQTT_BROKER = "MQTT Broker IP or DNS Name"
MQTT_PORT = 1883
MQTT_KEEPALIVE_INTERVAL = 45
MQTT_TOPIC = "testTopic"


# Define on_connect event Handler
def on_connect(mosq, obj, rc):
        #Subscribe to a the Topic
        mqttc.subscribe(MQTT_TOPIC, 0)

# Define on_subscribe event Handler
def on_subscribe(mosq, obj, mid, granted_qos):
    print "Subscribed to MQTT Topic"

# Define on_message event Handler
def on_message(mosq, obj, msg):
        print msg.payload

# Initiate MQTT Client
mqttc = mqtt.Client()

# Register Event Handlers
mqttc.on_message = on_message
mqttc.on_connect = on_connect
mqttc.on_subscribe = on_subscribe

# Connect with MQTT Broker
mqttc.connect(MQTT_BROKER, MQTT_PORT, MQTT_KEEPALIVE_INTERVAL )

# Continue the network loop
mqttc.loop_forever()
```

MQTT Client Application:



Add a new Connection

Connection Details

Connection name: Eclipse MQTT

Connection color scheme: 🟩

Hostname: tcp:// e.g. iot.eclipse.org

Port: 1883

Client ID: lens_JWBAO1iX8aCaTJGURrr7n41PTSa

Session: ☑ Clean Session

Automatic Connection: ☑ Automatic Connection

Keep Alive: 120 seconds

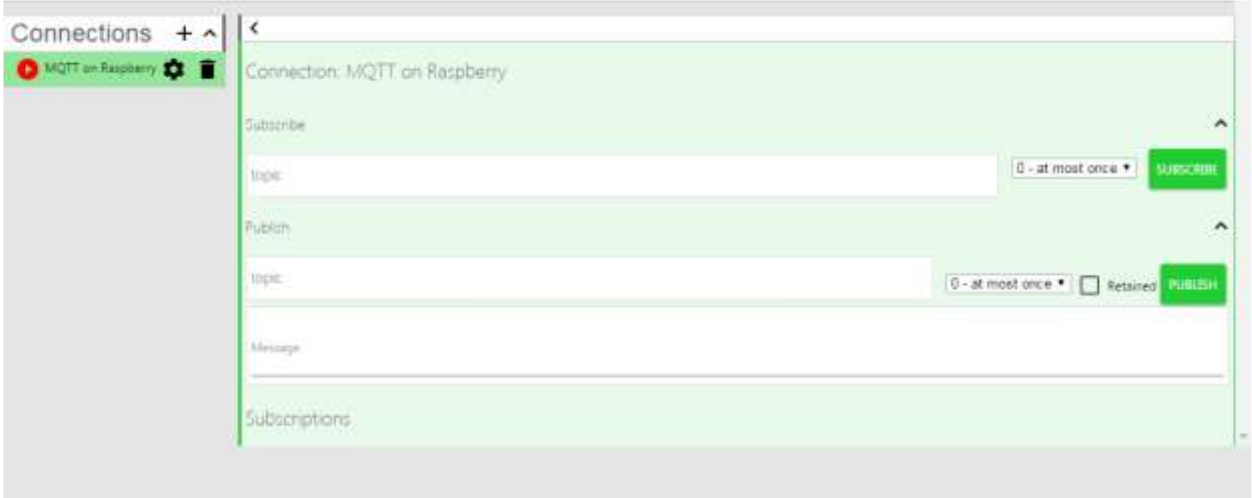
Credentials

Username: Enter username

Password: Enter password

Last-Will

Give any connection name, Host name shall be the IP address of MQTT broker (Raspberry pi). Select port 1883 and TCP connection, click create connection. Ensure that MQTT lens is connected to the Broker (Green color "pause" symbol in the left side of window)



in subscribe text box input the topic name and click subscribe button. Similarly give the same topic in publish - topic text box, type the message and click publish.

Above publisher and subscriber scripts can be used for M2M communication over MQTT protocol. Modify the scripts according to the application.