International Journal of Pure and Applied Mathematics
Volume 116 No. 6 2017, 161-166
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

IMPLEMENTATION OF MQTT PROTOCOL ON LOW RESOURCED EMBEDDED NETWORK

P Gopi Krishna[1], K Sreenivasa Ravi[2], V.S.S Sailendra Kumar[3], M.V.S.N Sai Kumar[4]
[1]*Research Scholar, Dept. of ECE, K L University, Guntur, A.P. India*
[1]gopikrishna.popuri@gmail.com
[2]Professor, Dept. of ECM, K L university, Guntur, A.P. India
[3,4]U. G student, Dept. of ECM, K L university, Guntur, A.P. India

Abstract: This paper discusses regarding the implementation of MQTT protocol on low resourced embedded network elements such as sensor node and sensor network node gateway. Dealing with IoT (internet of things), data collection is the primary objective and this is done through wireless sensor node network and the gateways, these operate on low processing speed footprints and low bandwidth wireless communication channels, even the gateways which are used as servers ought to be cost effective. So, the protocol used above the tcpie application layer protocol should be the light weight protocol and should have less overhead to cut back the processing burden on the embedded systems. So, here MQTT protocol which relies on publish/subscribe paradigm comes handy. Here in this paper we are using Node MCU as sensor nodes and Raspberry pi 3 as gateway for sensor node network gateway and mosquitto as MQTT broker on Raspberry pi and MQTT dashboard app as client to monitor data.

Keywords: IoT, HTTP, MQTT, Node MCU

1. Introduction

Internet of things (IoT) is a field which deals with connecting of real world physical devices like vehicles, industries, building, cities, things to internet for sensing of the real-world parameters and transmitting those to remote locations for analysis, estimation and future prediction [1]

Applications of IoT are endless and it has been into many fields like military [2], medical [2] and even it helped to solve the environmental problems for example microsoft's eye on earth project helped many European countries to check the quality of water and air [2]

In context with sensor nodes deployed in wide area the networks to which they connect should be of wide range and building such networks involves with heavy cost so many times. An existing well established wide GSM should be used to cope up with the cost of project. This

IoT deals with three things

I. Sensing and local processing of data
II. Transmission of data to the remote server
III. Analysis of data done at server

1.1 Transmission of data in IoT

While coming to the transmission of data in the internet of things it is done in many ways but for wide area of sensor deployment sensor node should be equipped with Tcp/ip stack which eases the communication of this sensor node to host so while using tcp/ip stack an application layer should also be used in conjunction to facilitate the formatting of data .

Challenges in transmission of data in IoT has lot of problems in the part of transmission by using existing architecture which for connecting large computers to the network and the challenges are[3]

1.1.1 Low processing power and resources

As the sensor nodes are small microcontroller having less processing power and low resources like less ram, low memory footprints and power constraints (as they are battery powered) they can't be connected to existing internet environment right away certain modifications should be done to optimize their connection to internet

1.1.2 Unreliable networks

network architecture is efficient enough in human to human communication and human to machine communication but this network is not reliable enough to (M2M) communication. In order to use this network effectively a method should be used to increase the reliability while connecting to this type of unreliable networks [2][4][5].

1.1.3 Security

As billions of devices are connecting to internet for the transfer of data there is lot of possibility that the data can be hacked for malicious purposes so data coming from the sensor nodes should be well protected and this should be done by a protocol which is light enough to run on the sensor nodes [4]

1.2 Addressing to the problems

The above mentioned three problems can be addressed easily without a lot of change in the architecture only when the they are handled by the application layer which on the top of the TCP/IP stack

In single line to say the application layer protocol which should be used in the in context with IoT should be, a lightweight protocol (in terms of execution), should be able operate over unreliable networks and should have high security

1.3 Disadvantages of Http protocol

The widely existing application layer protocol is HTTP (Hyper Text Transfer Protocol) which when is used in context has certain advantages but its disadvantages overruled the advantages provided by it. So, following are the disadvantages [6].

1.3.1 Request/Response paradigm

The paradigm used by HTTP protocol is a request/response paradigm And when this is operated the over the low number of devices can be efficient but when this operated over large number of devices it is going to flood the network and this going to create congestion This paradigm also cannot handle the burst throughput [6][7].

1.3.2 Large overhead

As this protocol is designed to transfer the text, images and different types of data formats it is very much needed to specify different things about the data in head of the message. This inevitably increases the size of the head in the message But in IoT it mostly deals with the transmission of only sensor data so it need not specify about the data as it is going to be in the form of only text [7]

All the above-mentioned challenges in IoT can be addressed by using a lightweight application layer protocol MQTT.

1.4 Overview on MQTT protocol

MQTT is a open source stand for M2M (Machine to Machine) communication this protocol is designed by IBM MQTT stands for Message Queue Transport Telemetry protocol. It is a application layer protocol it is a lightweight protocol as it does not include large overheads

1.4.1 Advantages MQTT of protocol

I. The paradigm it uses is publish/subscribe paradigm which gives it low overhead and makes this a lightweight protocol [8][9]

II. As it uses a broker to transfer the messages between client's authentication can be implemented on the broker which gives security [9] to the data

III. QoS [9] feature of the protocol helps it in handling the unreliable networks it is done by using different levels of Quality of Service for different types of messages

IV. It is also efficient at multicasting [9] of the messages

V. SSL encryption can also be used in conjunction with this protocol which helps in end to end encryption

1.4.2 Elements of protocol

MQTT uses the Publish/subscribe paradigm [10] [11] in contrast with traditional client-server model which uses Request/response paradigm here the messages sent from sender are not directly sent to the receiver instead it is routed to the receiver by an intermediate network element which also queues the messages needed to send to certain receiver in this process. The sender doesn't have any information about receiver and

this is done using "concept of topics"

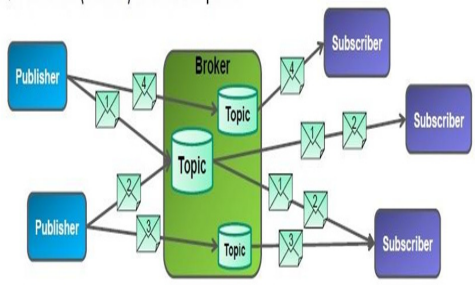


Figure 1. Example of MQTT network

In this protocol the as the sender doesn't have any information about the receiver the only link between the sender and receiver is the variable known as "topic" [10] to which the sender sends the messages and receiver receives the messages which resides in the software present in the server

In MQTT protocol the network element which sends messages to the topic is known as publisher and the receiver which receives the messages is known as subscriber.

The software in the server which facilitates the communication between the publisher and subscriber is known as Broker [10] [11].

To send a message to the sender publisher needs to connect to the server with a username and password and then create a topic with name and send data and to receive data of certain topic the subscriber should also get authorized with username and password and subscribe to the topic The priority to the message can be set by setting different levels of QoS[10][11]

Data transfer using HTML and JSON is shown in the paper [12] and data transfer using MQTT and pre-established server in the cloud is shown in the paper [13] so here in this paper we are showing the implementation of MQTT protocol on low resourced embedded systems in the intranet for the data transfer

2. Experimentation

Here we are using NodeMCU as the sensor node and Raspberry pi as server with MQTT Broker and an android phone with App to monitor data

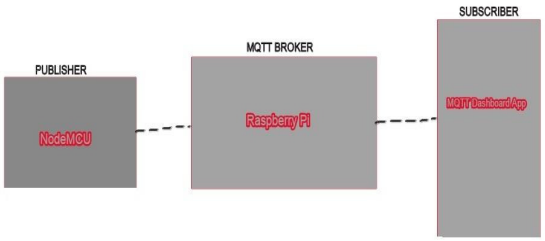


Figure 2. Block diagram of MQTT network

2.1 NodeMCU:

The open source firmware which is built in conjunction with ESP8266 Wi-Fi SoC is known as the NodeMCU and it can be programmed by using Customized Lua scripting language known as NodeLua script it can be programed by burning the rom prebuilt firmware and calling the functions by sending the NodeLua script program or by using Arduino IDE the total program can be upload with necessary functions.

Here in this project we are using this NodeMCU as the sensor node to send the data it contains Wi-Fi interface and GPIOs in microcontroller on the SoC can be used for general purpose as well, so our sensors and actuators can be directly connected to NodeMCU. It has 1 analog pin and 8 digital GPIO pins, which server's different purposes like UART, PWM, I2C etc.

NodeMCU can be programed by using Arduino IDE which provides different libraries through it. NodeMCU can be readily connected to the Wi-Fi network WiFi.begin(ssid, password) is used to connect to Wi-Fi network when provided with ssid and password of the network WiFi.status() gives the status of the connection WiFi.localIP() gives the local ip the sensor node.

NodeMCU can be connected to MQTT server by using client.functionsetServer(mqtt_server_ip, Port_no) ipaddress of the server and port number should be passed to this client.publish("Topicname", data) can be used to publish data to which topic name and data are passed ,client.subscribe("Topic_name") is used to subcribe to a topic on the broker

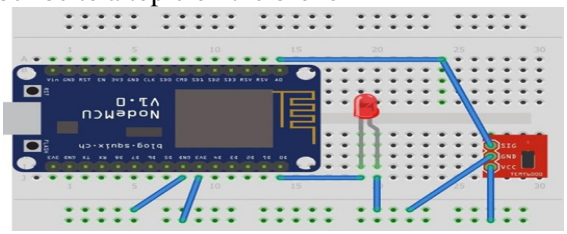


Figure3. Connections between NodeMCU, sensor and LED

As shown in the figure 3 LED is connected to D0(digital gpio pin) of NodeMCU and a light sensor is connected to the A0(Analog gpio pin).

This setup Publishes data of light sensor to the topic named light on the broker and it is also subscribed to the topic named "led_state" on the broker which helps the led on this setup remotely operated by client connected to the broker.

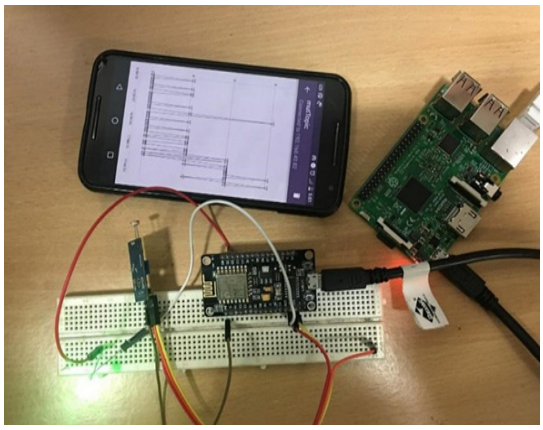


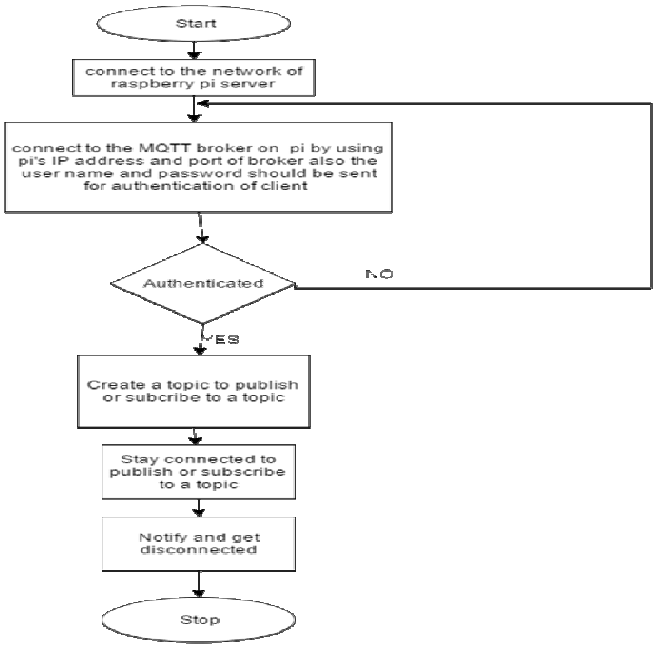
Figure 4. Picture showing total experimental setup

2.2 Raspberry pi:

Raspberry Pi is an open source single board computers it uses a Broadcom BCM2837 SoC with a 1.2 GHz 64-bit quad-core ARM Cortex-A53 processor, with 512 KB shared L2 cache [14].and run on various versions of Linux OS.

Here we are using this Raspberry pi running raspbian OS as server or gateway to facilitate MQTT network a open source MQTT broker software Mosquitto[15] is installed on the Raspberry pi this can be done by using command "sudo apt-get install mosquito" [15] and by using series of commands in the Terminal of Raspbian OS username and password can be set and when everything is set Mosquitto broker can be started by using command "mosquitto -c /etc/mosquitto/mosquitto.conf" [15]

And by following this sequence the client can be connected to the broker to publish or subscribe to topic



Flowchart 1: Flowchart showing procedure connection establishment and data transfer between client and server

3. Results

Mobile App:

MQTT Dashboard is a mobile app which runs on a Android mobile phone which can be used for monitoring data sent from sensor node through MQTT protocol it provides user with graphical representation of data.

This app can be connected to the MQTT broker by entering the credentials of the Broker i.e IP address, port number, username and password. This acts as the client which receives the data from the NodeMCU i.e light sensor data, the led on NodeMCU is controlled by using this app and this can have done by Subscribing to the topic "mutTopic" and publishing to the "inTopic"

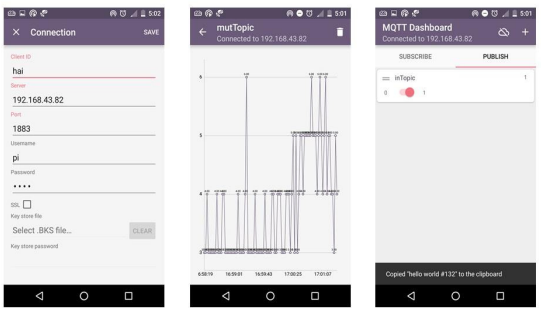


Figure5. Screenshots showing data on mobile application.

4. Conclusion & future scope:

In this paper, we have discussed about the implementation of MQTT protocol on low resourced embedded systems, which means that both the clients and also server implementing the protocol are low resourced existing work implemented this protocol on low resourced clients but servers used are mainframe computers but here the server we have used also a single board computer with low specifications, and this type of setup can be used where cost, space of the project are crucial. For example, we can use this in-home automation domain and many applications where main frame computers can't be used as server in the network. As here broker is the main network element in the network any failure in this leads to collapsing of total network of sensors, and it is pretty much possible for a low specs device to fail. So, this work can be extended by implementing effective fault tolerance and load balancing techniques by running more than one single board computers in parallel in the network.

Acknowledgments

Authors sincere thanks to K L University for providing research laboratories sponsored by DST-FIST.

References

[1] The internet of things: A survey L Atzori, A Iera, G Morabito - Computer networks, 2010, Elsevier

[2]Future Internet: The Internet of Things Architecture,Possible Applications and Key Challenges Rafiullah Khan ∗ , Sarmad Ullah Khan † , RifaqatZaheer ‡ and Shahid Khan 2012 10th International Conference on Frontiers of Information Technology

[3]Yen-Kuang Chen, "Challenges and Opportunities of Internet of Things", in the proceedings of 17th Asia and South Pacific Design Automation Conference, 30 Jan.-02 Feb., 2012, Santa Clara, CA, USA.

[4] Identities in the Future Internet of Things Sarma, A.C. &Girão, J. Wireless PersCommun (2009) 49: 353. doi:10.1007/s11277-009-9697-0

[5] Miorandi, D., Sicari, S., De Pellegrini, F., &Chlamtac, I. (2012). Internet of things: vision, applications and research challenges. Ad hoc Networks, 10(7), 1497–1516.

[6]Analysis of HTTP Performance Joe Touch, John Heidemann, and Katia Obraczka Aug. 16, 1996 USC / Information Sciences Institute

[7]HTTP2 VS HTTP1.x (Specifications&Suggestions) Girish K. M. ,Akshay R. Muntode International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 4 Issue 5, May 2015.

[8] A Secure and Fast Authentication implementation between the Entities using Trust Aware Algorithm, P. SIVA SANKAR, International Innovative Research Journal of Engineering and Technology, volume 2, Issue 1, September 2016.

[9] MQTT For Sensor Networks (MQTT-SN) Protocol Specification Version 1.2 Andy Stanford-Clark and Hong Linh Truong November 14, 2013 Copyright Notice

[10] Internet of Things application layer protocol analysis over error and delay prone links Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop (ASMS/SPSC), 2014 7th

[11] A. Stanford-Clark and H. L. Truong, MQTT for sensor networks(MQTTs), http://www.mqtt.org/MQTTs Specification V1.0.pdf, Oct. 2007.

[12] "MQ Telemetry Transport," http://mqtt.org.

[13]"From the Internet of Things to the Web of Things: Resource-oriented Architecture and Best Practices" , Book Title:Architecting the Internet of Things, Pages pp 97-129, Copyright:2011, DOI:10.1007/978-3-642-19157-2_5, Print ISBN: 978-3-642-19156-5, Online ISBN: 978-3-642-19157-2, Publisher: Springer Berlin Heidelberg, Copyright Holder Springer-Verlag Berlin Heidelberg

[14] Security: Smart Homes Using Internet of Things (IOT) #1Rugved Amrutkar, #2Sanket Vikharankar, #3Lochan Ahire International Engineering Research Journal (IERJ) Volume 2 Issue 2 Page 558-561, 2016, ISSN 2395-1621

[15]"Raspberry Pi "https://en.wikipedia.org/wiki/Raspberry_Pi

[16] "Mosquitto An Open Source MQTT v3.1/v3.1.1 Broker" ,https://mosquitto.org/