Journal of Advanced Research in Technology and Management Sciences (JARTMS)
Volume: 02 Issue: 02 ISSN: 2582-3078 March- April 2020
Available online at: http:// www.jartms.org

Login based Home Automation System using MQTT Protocol

Prof. V. Muralikrishna[1], Rahaman Sk[2], Sai Pavan Kumar S[3], N.V.ViswaTeja[4]

Department of Elecronics and Communication Engineering,
KKR & KSR Institute of Technology and Sciences GUNTUR 522003 INDIA.
veerepalli75@gmail.com

Abstract— As we know in general home automation we generally use an android app which directly connects to the nodes via rooter (PAN) connection. Almost every home automation service will not provide password security in these days. To overcome these security issues we are designing a login based home automation service with the help of raspberry pi as a server. This raspberry pi stores the information of the user and stores their data like name, user ID, password, etc, in corresponding data base. Here we use HTML, JavaScript and CSS for handling web applications. Here we are using several node MCU boards as subscribers where raspberry pi acts as a publisher device. Here raspberry pi can act as a Gateway between wireless sensor network and Internet. To connect devices between host to slaves here we are using MQTT protocol. Where MQTT protocol works as a public and subscribe model approach. In MQTT protocol server acts as a broker between clients. Due to presence of broker data gets more secure. To host a web page from raspberry pi we are using apache 2software to host the web page.

Keywords: Message Queuing Telemetry Transport (MQTT), Node MCU, Mosquitto, Login based Home automation, Node-Red, HTML, CSS, JS, Apache, Raspberry pi.

* Correspondence Author

Prof. V. Muralikrishna[1]
Rahaman Sk[2]
Sai Pavan Kumar S[3]
N.V.ViswaTeja[4]
Department of Elecronics and Communication Engineering,
KKR & KSR Institute of Technology and Sciences GUNTUR 522003 INDIA.
veerepalli75@gmail.com

Login based Home Automation System using MQTT Protocol

I. INTRODUCTION

The Internet of Things (IoT) is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.The definition of the Internet of Things has evolved due to the convergence of multiple technologies, real-time analytics, machine learning , commodity sensors, and embedded systems. Traditional fields of embedded systems, wireless sensor works, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of Things. In the consumer market, IoT technology is most synonymous with products pertaining to the concept of the "smart home", covering devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smartphones and smart speakers.

II. LOGIN PAGE

The login page allows a user to gain access to an application by entering their username and password or by authenticating using a social media login. The Login page is created using programming Languages like HTML (Hyper Text Markup language), CSS (Cascaded Style Sheets) and JS (Java Script). JavaScript runs in HTML on the web browser at client side. JavaScript is used for client side validation of the form. Our login page Includes Input text for entering user name, input text for entering password, Login button to submit form data to the server side program. In this Project we are not writing any server side program for validating it with database. If you want to validate the user with database you can write any of the server side programs such as JSP,Servletor PHP to validate the user against database.

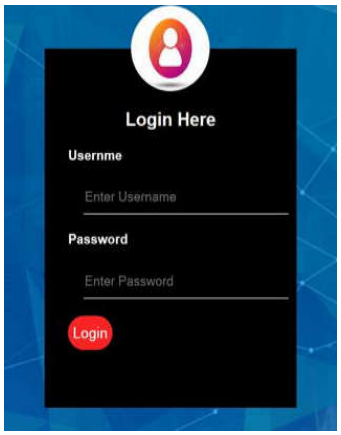

Fig. 1. Login page Format

III. MESSAGE QUEUING TELEMETRY TRANSPORT

Message Queuing Telemetry Transport (MQTT) is a light weight transport protocol that efficiently uses the network bandwidth with a 2 byte fixed header [1]. MQTT works on TCP and assures the delivery of messages from node to the server. Being a message oriented information exchange protocol, MQTT is ideally suited for the IoT nodeswhich have limited capabilities and resources. MQTT was initially devel-oped by IBM [2] in 1999 and recently has been recognized as standard by Organization for the Advancement of Structured Information Standards (OASIS) [3].MQTT is a publish/subscribe based protocol. Any MQTT connection typically involves two kinds of agents: MQTT clients and MQTT public broker or MQTT server.

Data that is being transported by MQTT is referred to as application message. Any device or program that is connected to the net- work and exchanges application messages through MQTT is called as an MQTT client. MQTT client can be either publisher or subscriber. A publisher publishes application messages and subscriber requests for the application messages. MQTT server is a device or program that interconnects the MQTT clients. It accepts and transmits the application messages among multiple clients connected to it. Devices such as sensors, mobiles etc. are considered as MQTT client. When an MQTT client has certain information to broadcast, it publishes the data to the MQTT broker. MQTT broker is responsible for data collection and organization. The application messages that are published by MQTT client is forwarded to other MQTT clients that subscribe to it. MQTT is designed to simplify the implementation on client by concentrating all the complexities at the broker. Publisher and subscriber are isolated, meaning they need not have to know the existence or application of other. Before transmitting the application messages, control packets are exchanged based on

the QoS associated with them. An MQTT control packet consists of a fixed header, a vari- able header and payload. CONNECT, CONNACK, PUBLISH, PUBACK, PUBREC, PUBREL, SUBSCRIBE, SUBACK, etc. are some of the MQTT control packets [4] exchanged between MQTT clients and MQTT server." Topic" in MQTT provide the routing information. Each topic has a topic name and topic levels associated with it. There may be multiple topic levels separated by / in a topic tree. Wildcard characters such as # and + are used to match multiple levels in a topic. Featuring the queuing system, MQTT server buffers all the messages if client is offline and delivers them to the client when the session is enabled.

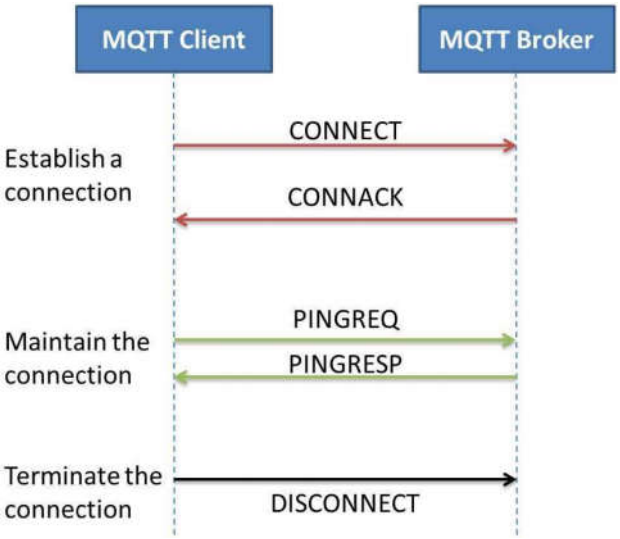

Fig. 2. Establishing, maintaining and terminating MQTT connection

A. Establishing a connection Upon the successful establishment of network between the MQTT client and the MQTT server, control packets are exchanged between the client and the server. The client that wishes to connect to the MQTT server sends a CONNECT packet to the server specifying its identifier, flags, protocol level and other fields. The server acknowledges the client with the specified identifier through CONNACK packet with a return code denoting the status of connection.

B. Publishing the application messages

If the client desires to be a publisher, it sends a PUBLISH packet to the server. This packet contains details about the QoS level of transmission, topic name, payload, etc. MQTT supports three levels of Quality of Service (QoS) [5] to the client. If the application messages are transmitted at QoS 0, the client does not receive any acknowledgment for the published packet. For QoS 1, the server acknowledges the published packet with PUBACK including the packet iden- tifier. However, in QoS 2, four packets are exchanged. The server acknowledges the receipt of PUBLISH packet with the PUBREC packet. MQTT client then sends a packet to release publish with a PUBREL packet. The server then sends the fourth packet PUBCOMP, indicating the completion of publishing the application message on the given topic.

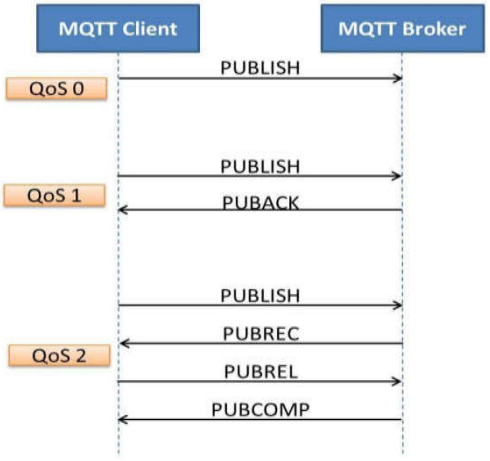

Fig. 3. Client publishing messages to the server with various QoS

C. Subscribing to a topic

If the MQTT client want to subscribe to the application messages published on topic, it sends the SUBSCRIBE packet along with the topic name indicated in UTF-8 encoding. The server acknowledges the subscription with SUBACK packet along with a return code denoting the status of request. Once the subscription is successful, the application messages on the specified topic are forwarded to the client with the maximum QoS. To unsubscribe a topic, the client sends an UNSUBSCRIBE packettotheserverwhich acknowledges it with the UNSUBACK packet.

D. Maintaining the connection alive

After a certain time-out, the connection between the client and the server is terminated. To maintain the connection, the client indicates that it is alive by transmitting a PINGREQ packet to the server. The MQTT server responds to the client with the indicated identifier with a PINGRESP packet and maintains the connection alive.

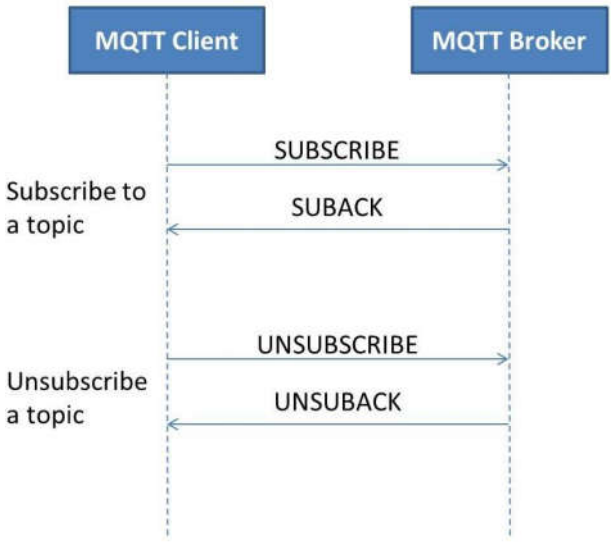

Fig. 4. Client subscribing and unsubscribing to the topic

E. Terminating theconnection

To terminate the connection, the MQTT client sends a DISCONNECT packet to the server. The server does not acknowledgethispacket.However,alltheapplicationmessages related to the client will be flushed off and the client is disconnected from the server.

IV. NODE RED

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

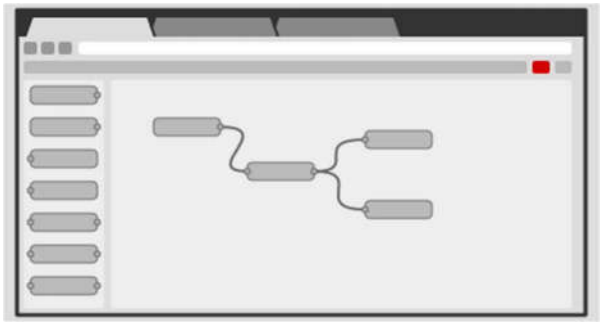

Fig. 5. Node RED Representaytion

Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette. Flows can be then deployed to the runtime in a single-click.JavaScript functions can be created within the editor using a rich text editor.A built-in library allows you to save useful functions, templates or flows for re-use.

V. IMPLEMENTATION DETAIL

A. Network setup

The intensity of light is sensed using LDR sensor connected to ESP8266 development board. ESP8266 development board processes the sensor data and performs actuation. It acts as a gateway for data transmission through WiFi. ESP8266 is configured as MQTT client publishing the sensor data to the MQTT broker and subscribing for the commands to control the actuation. LED and buzzer is used as actuators in the prototype. ESP8266 module publishes the sensor data under the topic ′esp\sense′. It subscribes for the topic ′esp\led′ and ′esp\buzzer′ to receive commands to control LED and buzzer connected to the GPIOs of ESP8266. MQTT mosquito broker is set up for ESP8266 to publish and subscribe to the application messages. Other MQTT clients such as PCs and Mobiles can connect to MQTT server through existing communication technologies such as Ethernet, 2G, 3G, WiFi etc.

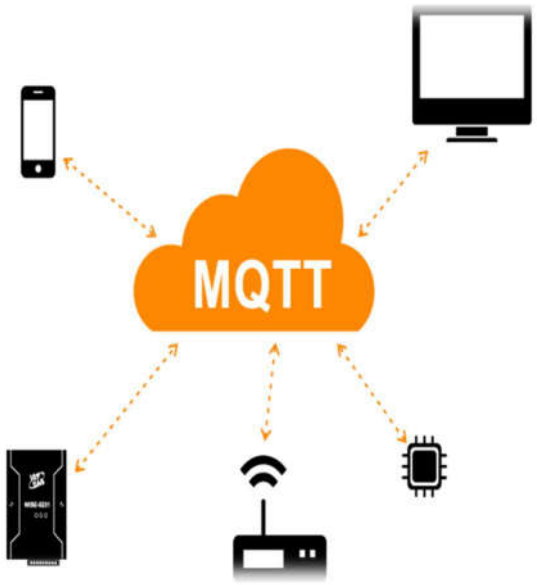

Fig. 6. Network Connection

B. NODE MCU

ESP8266 [9] is a low cost development board that consol-idates GPIOs, I2C, UART, ADC, PWM and WiFi for rapid

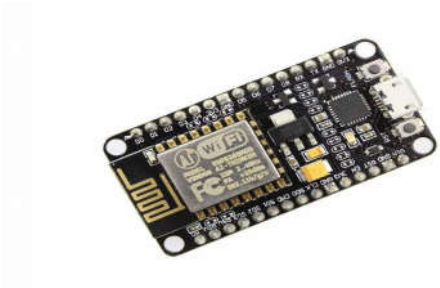

Fig. 7. NODE MCU

C. Server setup

Apache is a popular web server application you can install on the Raspberry Pi to allow it to serve web pages.
On its own, Apache can serve HTML files over HTTP. With additional modules it can serve dynamic web pages using scripting languages such as PHP.

ToInstallApache,openaterminalwindowby selecting Accessories > Terminal from the menu.Install the apache2 package by typing the following command into the terminal and pressing Enter:

sudo apt-get install apache2 -y

By default, Apache puts a test HTML file in the web folder that you will be able to view from your Pi or another computer on your network.
Open the Apache default web page on your Raspberry Pi: Open Chromium by selecting Internet > Chromium Web Browser from the menu.

Enter the address http://localhost.

You will also be able to open this web page from any other computer on your network using the IP address of your Raspberry Pi, e.g. http://192.168.1.10.To find out your Raspberry Pi's IP address, type hostname -I into the terminal window. Your Raspberry Pi's IP address is a really useful and will allow you to remotely access it.

VI.RESULTS AND DISCUSSION

MQTT server is set-up using Mosquitto. Upon setting up of the server and starting the service, the pub/sub of applica- tion messages can be viewed on the command prompt. Any authorized MQTT client can publish or subscribe to the data onto this server with the ID of its host IP and port 1883. The sensor data is aggregated by ESP8266 module and published to the MQTT broker on the topic ´ esp/sense ´ . Any MQTT client. subscribed to this topic can view the sensor readings. LED and Buzzer connected to ESP8266 can be turned ON and OFF by publishing suitable control commands on appropriate topics. The exchanged MQTT messages are grasped through MQTTLens and MyMQTT android application. Multiple actuators can be controlled by using wild cards (+ or #) at the same time.

VII.CONCLUSION AND FUTURE WORK

MQTT is thus a light weight protocol that occupies low bandwidth and consumes less power. Considering the ease of wireless internet access through WiFi, MQTT client ap-plication is built on ESP8266. A prototype of MQTT based home automation system is implemented on ESP8266. The sensors and actuators connected to ESP8266 are remotely monitored and controlled through a common home gateway. Thus the existing infrastructure can be used to enhance the home appliances and make them smart. This implementation provides an intelligent, comfortable and energy efficient home automation system. It also assists the old and differently abled persons to control the appliances in their home in a better and easier way.Taking this further ahead, cloud platform can be used to aggregate, analyze and visualize data. Customized GUI can be developed to remotely access the devices to monitor and control them.

VIII. REFERENCES

[1]Mqttv3.1 protocolspecification.[Online].Available:
http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html [2] Hivemq. [Online]. Available:
http://www.hivemq.com/blog/ mqtt-essentials-part-1-introducing-mqtt
[3] Mqtt version 3.1.1 becomes an oasis standard.[On- line]. Available:
https://www.oasis-open.org/news/announcements/ mqtt- version-3-1-1-becomes-an-oasis-standard
[4] (2014, October) Mqtt version 3.1.1 oasis standard. [Online]. Available:
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html
[5] Oasis mqtt version 3.1.1. [Online]. Available: http://docs.oasis-open. org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html
[6]H. ElKamchouchiandA. ElShafee, "Design andprototypeimplemen-tation of sms based home automation system," in Electronics Design, Systems and Applications (ICEDSA), 2012 IEEE International Confer- ence on, Nov 2012, pp. 162–167.
[7] S. Nasrin and P. J. Radcliffe, "Novel protocol enables diy home au-tomation," in Telecommunication Networks and Applications Conference (ATNAC), 2014 Australasian, Nov 2014, pp. 212–216.
[8] Eclipse. Mosquitto an open source mqtt v3.1/v3.1.1 broker.[Online]. Available: http://mosquitto.org/
[9] Nodemcu – an open-source firmware based on esp8266 wifi-soc. [Online]. Available:http://nodemcu.com/index en.html/