INVESTIGATION ON NETWORK PERFORMANCE FOR NOTIFICATION ALERT SYSTEM USING MQTT PROTOCOL

MUHAMMAD KHAIRUL AMRIE BIN MOHD DIN

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

INVESTIGATION ON THE NETWORK PERFORMANCE FOR NOTIFICATION ALERT SYSTEM USING MQTT PROTOCOL

MUHAMMAD KHAIRUL AMRIE BIN MOHD DIN

This report is submitted in partial fulfilment of the requirements for the degree of Bachelor of Electronic Engineering with Honours

Faculty of Electronic and Computer Engineering
Universiti Teknikal Malaysia Melaka

2020

DECLARATION

I declare that this report entitled "INVESTIGATION ON NETWORK PERFORMANCE FOR NOTIFICATION ALERT SYSTEM USING MQTT" is the result of my own work except for quotes as cited in the references.

Signature : ……

Author : Muhammad Khairul Amrie bin Mohd Din ……

Date : 26 June 2020 ……

APPROVAL

I hereby declare that I have read this thesis and in my opinion this thesis is sufficient in terms of scope and quality for the award of Bachelor of Electronic Engineering with Honours.

Signature : .....

Supervisor Name : Dr Juwita Mohd Sultan

Date : 29 Jun 2020

DEDICATION

To my parents, Mohd Din Bin Abdullah and Kamariah Binti Yusoff. Thank you for love and support

ABSTRACT

The paradigm of the Internet of Things (IoT) anticipates a world where everything is connected, and can be monitored and controlled remotely. From woodlands to factories, if relevant data are collected and analysed, we can improve efficiency and reduce costs. Additionally, remote control of our homes may drive a new wave of gains in energy efficiency. Moreover, we need machines that can operate on batteries for years in order to connect anything, and this also needs protocol enhancement. Internet of Things (IoT) application layer protocols are becoming increasingly popular in a wide range of scenarios where low-cost , low-power or resource-constrained devices are present. The most diffused protocols are MQTT. MQTT is a messaging protocol designed to be lightweight and built on top of the Transmission Control Protocol (TCP). In addition, MQTT allows offline messaging to handle clients that are disconnected.

ABSTRAK

Paradigma Internet of Things (IoT) menjangkakan dunia di mana semuanya terhubung, dan dapat dipantau dan dikendalikan dari jarak jauh. Dari kawasan hutan hingga kilang, jika data yang relevan dikumpulkan dan dianalisis, kita dapat meningkatkan kecekapan dan mengurangkan kos. Selain itu, kawalan jauh kediaman kita dapat mendorong peningkatan gelombang kecekapan tenaga. . Lebih-lebih lagi, kita memerlukan mesin yang dapat beroperasi pada bateri selama bertahun-tahun untuk menghubungkan apa-apa, dan ini juga memerlukan peningkatan protokol. Protokol lapisan aplikasi Internet of Things (IoT) menjadi semakin popular dalam pelbagai senario di mana peranti kos rendah, kuasa rendah atau sumber terhad ada. Protokol yang paling banyak disebarkan adalah MQTT. MQTT adalah protokol pesanan yang dirancang agar ringan dan dibina di atas Transmission Control Protocol (TCP). Sebagai tambahan, MQTT membenarkan pemesejan luar talian untuk menangani klien yang terputus.

ACKNOWLEDGEMENTS

Alhamdulillah, I am grateful to the God for the good health and wellbeing that were necessary to complete this thesis report.

I wish to express my sincere thanks to Dr. Juwita Binti Mohd Sultan, Supervisor of my Final Year Project, for providing me with all necessary facilities for the research.

I place on record, my sincere thank you to my parents for the continuous encouragement.

I am also grateful to lecturers in the Faculty of FKEKK. I am extremely thankful and indebted to them for sharing expertise, and sincere and valuable guidance and encouragement extended to me throughout my years in the faculty.

I take this opportunity to express gratitude to all my friends and seniors for their help and support. I also thank my family for the unceasing encouragement, support and attention. I am also grateful to my fellow housemates who support me through this venture.

I am place on record, my sense of gratitude to one and all, who directly and indirectly, have lent their hand on this venture.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Ethernet protocol was intended to give a correspondence organize that can interface numerous PCs to an outside data sharing transport. Ethernet is commonly utilized for the arrangement of neighborhood (LANs). Ethernet identify these associations in machines, and utilize an irregular occasions to keep away from the following occurrence. Ethernet faces specific issues as apportioned transfer speed and time allotments are missing for use in correspondences frameworks. An Ethernet organize makes extra complexities since when utilized with remote, the measure of transmission capacity accessible can change [1]. A large number of these system execution upgrade innovations are utilized to improve organize execution between at least two areas, where connection holding/conglomeration is given at least one of those areas. While the reinforced/accumulated connections give noteworthy

improvement in arrange execution over the associations accessible to convey organize traffic to a passage on a system's spine [2].

Alert notification systems are especially helpful regarding army bases, Schools, enterprises and industry offices and different associations that perform on frameworks and permitting the client or focal interchanges frameworks to properly give information to a majority of clients or dynamic clients including basic crisis information or different news alarms. The contact can be guided to determined subsets of clients who buy in to the notice program. [3] The current innovation, in one structure, considers an individual security cautioning framework utilizing a large number of cell phones in remote correspondence with a system. A server, connected to the system, collaborates with every cell phone through which the server is connected to a database of individual security subtleties identified with predefined alarms/occasions [4]. The information assortment operator must have an interface to the system pile of the working framework, and will choose much of the time which applications utilize the Stack organize. The information assortment specialist may along these lines track data concerning a system association, the length of the association, just as any applications and frameworks occupied with the association. [5]

Wireless device is remote broadcast communications cation arrange intended to transmit and get voice and information by means of remote correspondence beneficiaries, for example, cell base stations, WIFI centers, cell towers, satellite systems, and so on between remote gadget 10 and at least one getting gadgets. Any cutting edge media transmission framework can be utilized to transmit information as well as voice information in the innovation [6].

For quite a while the Internet of Things (IoT) is getting mainstream. Basic and light gadgets furnished with sensors and remote correspondence capacities are very utilized in numerous application situations, for example, remote sensor systems, ecological checking, e-wellbeing, etc. Notwithstanding the particular circumstance, comparative necessities apply to all IoT usage: organize gadgets work with low-transmission capacity remote transmitters and recipients to transmit/get information from a fundamental information concentrator (sink hub) or other IoT hubs. [7]

MQTT is certainly one of the current advancements that has picked up the best exposure over the most recent couple of years, being essentially the standard true in both M2M and IoT applications. In any case, MQTT is turning into the most widely recognized convention for interfacing asset limited gadgets to significant cloud stages (e.g., Amazon AWS, Microsoft Azure, IBM Watson), which are all noteworthy their administrations through MQTT. The purposes behind such prominence come from MQTT 's amazing client side straightforwardness, which suits pleasantly in asset compelled applications, however bolsters dependability and different degrees of administration quality (QoS). [8] [9]

The convention runs over TCP/IP, or other system conventions giving arranged, lossless, bidirectional associations. "MQ Telemetry Transport (MQTT) is a lightweight intermediary based distribute/buy in informing convention intended to be open, basic, lightweight and simple to execute,". It permits information move of the style of telemetry which is only information of the sensor and the actuator. The sensors and actuators convey by means of MQTT message agent to applications. The center segments of MQTT incorporate customers, servers, dealers, meetings, memberships, and themes. The informing model distribute/buy in comprises of various distributers

and endorsers who are associated with a specialist. Distributers send (distribute) messages about a particular "point" to the specialist Subscribers register (buy in) with the dealer about their enthusiasm for specific subjects. All communication happens through a meeting between a server and customers. The spec depicts its messages and frameworks, as well. MQTT works over TCP/IP. Other than the guaranteed conveyance by means of TCP/IP, MQTT includes 3 more QoS layers top of TCP, conveyance at most once, conveyance at any rate once and conveyance precisely once. [10] [11] [12]

MQTT is a numerous to-numerous correspondence convention, it utilizes a focal representative to move messages between different gadgets. MQTT applications with a seemingly perpetual active TCP association connected to the dealer, this association was at first light on prerequisite. To assist gadgets with perceiving the imprint, MQTT doesn't permit data or type message naming. In MQTT the message types ought to tell all gadgets ahead of time with the goal that correspondence is conceivable. [13] [14]

A portion of the principle advantages of point divulgence is that moving or moving the assortment of system execution information to a system administrator 's clients empowers the system administrator to see organize data more effectively and in more prominent detail than customary methodologies. Since endorsers detailing an occurrence are encountering the system issue, supporters may report time points of interest, area, episode data, related episodes, and occurrence conditions that are troublesome if not difficult to repeat by a system engineer sent to the site to react to the announced system execution episode. [15]

MQTT (Message Queuing Telemetry Transport) is a texting convention dependent on agents being distributed/bought in. Worked to be anything but difficult to execute,

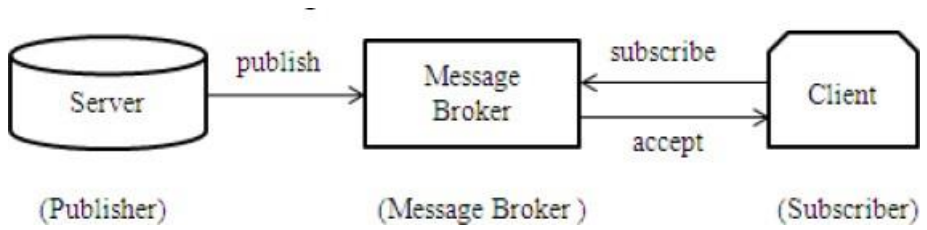
straightforward, lightweight and modest. The methodology is custom-made to the development of cell phones. Utilizing the webservice message dealer, the application recommends answers for the server to impart the expressed signs to the assigned cell phones and the versatile customers getting pushed alarms from the server With the ubiquity of the cell phone, individuals are progressively OK with cell phones to get ongoing messages, so texting is especially significant. The point of this task is to utilize Internet, to go to business clients the different fields of information, for example, science innovation , account, sports , diversion. [16] [17]

A distributer should initially sign in to the intermediary to impart. After positive contact indicated by an answer from merchant, the distributer should then enroll itself to the intermediary. The point and message is discharged to the merchant after viable enrollment by the client. A distinction bundle is sent to facilitate to end the meeting. On the off chance that there are no distinction bundles or any parcels got by the dealer, after a predefined time the distributer will be esteemed detached. Enlistment just happens once. Resulting join just require an associate bundle followed by the distribute message. [18]

Transmission of messages between different devices is necessary because an IoT appliance has to provide instructions for controlling a system to another appliance. Push protocol is the appropriate message communication protocol for IoT appliances compared to polling protocol, as it is built in poor bandwidth network. Via these push message services, MQTT, XMPP and CoAP protocols were introduced. Those protocols are applicable in different circumstances. In particular, MQTT was used as part of several IoT gadgets and instant message delivery systems, since it was intended to act as a lightweight protocol on low-power machines. [19].

IBM creates MQTT. It is both an instant messaging protocol and a publishing / subscribing messaging protocol based on the lightweight broker This is generally utilized in the field of portable pushing messages, because of its ease of use and versatility [22]. The persevering procedure to interconnection is thusly a progressively perfect arrangement, due for its undeniable potential benefits, MQTT convention becomes todays first decision. It's an open, simple, lightweight, and simple informing convention to execute. At first intended to interface huge amounts of remote gadgets and specialized instruments. The Protocol applies to a wide assortment of implicit gadgets. Wellbeing offices utilize this framework to speak with pacemaker suppliers and other clinical hardware wholesalers. It has been utilized by oil and gas industry to screen the gigantic separations off the oil pipeline.. [20] [21]

Publishers send a specific message topic to a message broker, subscribers subscribe to the message broker for similar news topics and the message broker-managed link between subscriber and publisher. The message is sent to subscriber when the message broker receives the posted messages. Publishing / subscribing messaging model enables multiple providers to publish messages on the same subject. It also allows multiple users to subscribe to messages about a subject. [23]



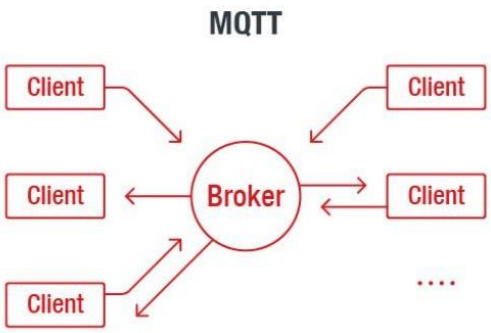


Figure 1.1 MQTT Basics

Client – The client is viewed as any distributer or endorser that interfaces with the brought together representative over a system. It is essential to take note of that MQTT is made out of servers and clients. Determined clients keep a representative meeting while the dealer doesn't screen the transient clients.

Broker: The Broker is primarily responsible for:

- All messages received
- Filter them
- Determine which one is interested
- Message to all subscribed customers
- Client Authentication and authorization

Topic:

A MQTT issue is a normalized various leveled string utilized for arranging and sending messages, which determines which message has been gotten by which client.

It fills in as the primary message distributing and dispersion center point of memberships.

1.1 Problem Statement

The motivation to create MQTT was to make a lightweight and transmission capacity proficient convention that would be empowered information with staggered administration quality help. MQTT was created as an amazingly light Publish/Subscribe for Messaging Transport. It is valuable for remote area associations where a little code impression is required, as well as premium system data transmission. There is some issue with utilizing the MQTT convention, that is:

- The client/subscriber must connect to the same topic as broker/publisher before they can communicate. This connection can be plain Transmission Control Protocol (TCP) or Internet Protocol (IP) connection or an encrypted Transport Layer Security (TLS). The solution is firstly the subscriber need to connect to the publisher and the publisher sends a message to a central topic which has multiple subscribers waiting to receive the message. One thing to know is the publishers and subscribers are autonomous, which means that they do not need to know the presence of each other.
- MQTT protocol are unsecure.. To solve this problem, the publisher need to consider having security built in and pay adequate attention to Internet of Things (IoT) security. Organization's teams should ensure that proper security mechanisms are in place when using protocols.

1.2 Objective

This study is generally aimed at investigating the MQTT protocol towards certain problem that have been faced in general life.

1. To identify and build a suitable cross-platform instant messaging/notification alert system using Arduino and WiFi using MQTT .
2. To analyze the network performance of the protocol such as delay, accuracy and stability based on the system.
3. The study will assist in evaluating the impact and effectiveness of this implementation protocol if apply to different and various type of real-life problem

1.3 Scope Of Project

The scope of this project will cover on the problems encountered by the fire station department. A few visits have been done in order to observe and analyze the real situation of the fire station. Literature works were prepared to obtain general ideas and better overview or enhancement from other researchers or somewhat similar projects done by them. The independent system will be produced by using MQTT protocol meanwhile the end user application will be design using Arduino. Ultimately, the outcome of the project will be analyze and presented in the final report.

CHAPTER 2

BACKGROUND STUDY

This project unfolds the background of MQTT Protocol. MQTT (Message Queuing Telemetry Transport) is an instant messaging protocol based on a broker. It is designed to be easy to implement, fast, lightweight and free. This is an identified method to drive the MQTT protocol-based notification system and their specification, Quality of Services and advantages.

2.1 Components

ESP 8266

The ESP8266 WiFi Module is a self-contained SOC that can provide any microcontroller with access to the WiFi network. The ESP8266 can either host an

application, or import from a separate application processor all Wi-Fi networking functions..

Figure 2.1 NodeMCU ESP82666

Resistor

The resistor is a passive electric component for generating resistance in the flow of electrical current. They are found in almost every electrical network and electronic circuit. Resistance is measured in ohms. An ohm is the resistance that occurs when a one-ampere current passes through a one-volt resistor over its terminals.

Figure 2.2 Resistor 1K Ohm

Light Sensor (LDR)

LDR stands for Light Dependent Resistor or Photoresistor, which is a passive electronic component, basically a resistor having a resistance that varies depending on the intensity of the light. A photoresistor is made of a high-resistance semiconductor that absorbs photons and gives bound electrons enough energy to jump into the conductive band based on the quantity and frequency of the photons absorbed by the semiconductor material. The resulting free electrons conduct electricity which results in reduced photoresistor resistance.

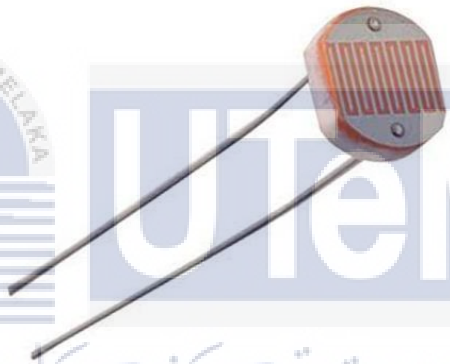


Figure 2.3 LDR

2.2 History of MQTT

Over the past decade , significant developments over communication and computer networks have allowed communication to be included in feedback in order to meet real-time requirements. It gave rise to a new concept for this development, which is Message Queuing Telemetry Transport (MQTT), which specializes in the study and design of control systems, where Machine to Machine (M2M) communication is

managed via a real time network. MQTT has gained growing attention in recent years because of their advantages such as low cost, easy maintenance and consistency.

Two physicists-Andy Stanford-Clark and Arlen Nipper-developed the protocol in 1999. They developed a remote oil pipeline monitoring system via unreliable satellite networks that required a low-power, easy to implement, and highly accurate messaging system.MQTT has grown to be a key messaging protocol used in many IoT systems since its development, and is ideal for home automation, hydroponics, or remote weather stations. [24]

2.3 Protocol Overview

The Message Queuing Telemetry Transport is a lightweight publish / subscribe protocol whose execution ideas are focused on the both end framework prerequisites and system resources, keeping up similarity and some level of nature of administration..

IBM and Eurotech developed MQTT as a very lightweight TCP / IP communication protocol for publishing / subscribing transmission of messages. Its publication and subscription are arranged according to the "topic" concept. MQTT allows devices to open a connection, keeps the connection open with low power, and delivers low overhead transport systems messages. Communication is very effective with low-power-constrained devices. [25]

Post / Subscribe Message Template. The design is based on the posting / subscription messaging template. Both the publisher and the recipient are clients in this model, the principle is called the destination post. By connecting to the message

broker, they transfer data across the Network. Publishers publish a common message subject to the message broker, subscribers subscribe to the message broker for specific news topics, and the message broker-managed connection between subscriber and publisher. [26]

The message will be sent to the recipient after receiving the published messages. Publishing / subscription model messaging enables multiple providers to post information on the same subject. It also allows multiple users to subscribe to messages about a subject. The message broker is then forwarded to various subscribers.

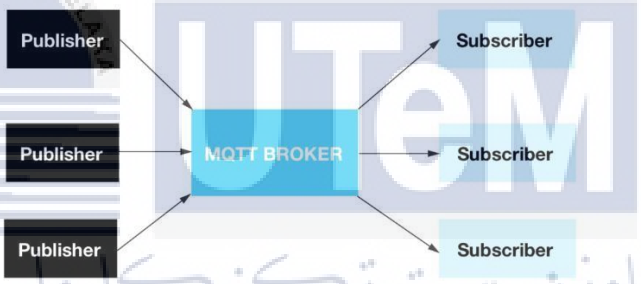


Figure 2.4 Overall Architecture Diagram

The European Telecommunications Design Institute (ETSI) is designing the ETSI Machine-to - Machine (M2M) architecture as a global M2M interface to integrate different vertical applications into a shared framework. ETSI sets a robust M2M architecture and the requisite interfaces to support end-to - end services. However, the ETSI M2M specification does not define how different types of protocols like MQTT can be supported. [27]

MQTT is a communications release / subscription protocol based on a broker. In view of the thought of "subject," it arranges its discharge and membership and all parcels are discharged by means of the dealer. A distributing subject ought to be determined with a certain goal in mind, however the point can be a membership special case that empowers MQTT customers to buy in to various themes without a moment's delay. A case of the MQTT is appeared in Figure 2.5. For this situation, Client A posts a message under the subject "animal/cat." Client B and Client C will get the message by means of the MQTT specialist as the two of them buy in to the subject "animal/cat". Then again, as it doesn't buy in to the "dog/pet" theme, Client D doesn't get the post.

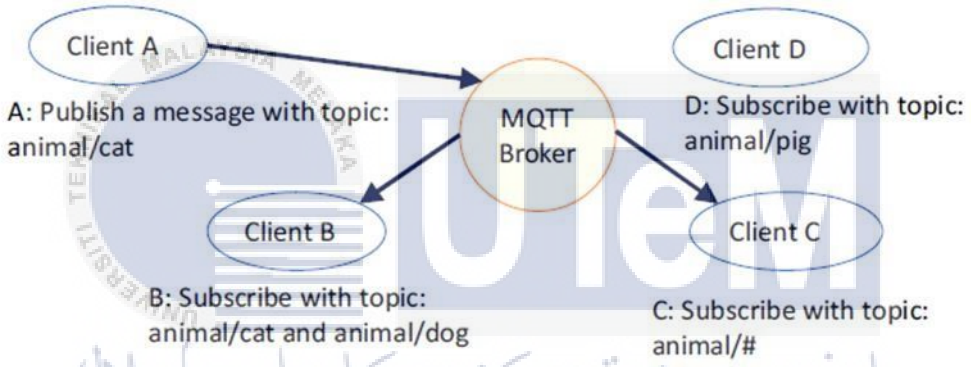


Figure 2.5 MQTT Example

2.3.1 MQTT Client

MQTT customer, not simply telephones, could be any IoT substance that sends or gets information. Customer can be any PC (e, g, microcontroller, server). MQTT customer type relies upon whether they are a supporter or a framework distributer.

2.3.2 MQTT Broker

The broker is a central tool that lists between the template being spoken and the hub. The MQTT broker's primary responsibility is to process the correspondence between MQTT clients and to distribute the messages among them based on their

topics of interest. After receiving the message, the broker will scan and find all the devices which have a subscription to this issue.

2.3.3 M2M Platform

OpenMTC is an M2M-platform compliant to ETSI. ETSI M2M's architecture consists of three domains as shown in Figure 2.6. Without the Application Service Capabilities Layer (DSCL) which has configured the Gateway Service Layer (GSCL), M2 M devices could be connected to the network domain via an M2 M gateway. The M2 M Network Domain links M2 M devices / gateways to M2 M applications using the standardized Network Service Functionality Layer (NSCL). The SCLs use an open interface set to expose M2M functionality.

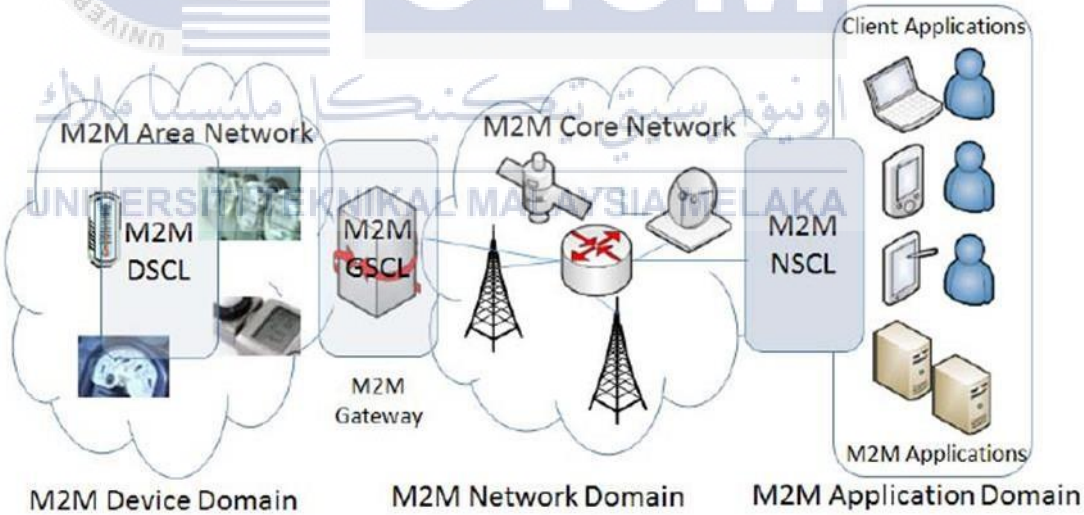


Figure 2.6 ETSI M2M Architecture

Image. Fig. 2.7 It indicates the message is flowing. Note that "MQTT Proxy" is specifically used as the application name, and "uniqueContainer" as the ID specific to the container: [27]

1) The MQTT Proxy requires that the GSCL compile a < application > with the "MQTT Proxy" ID on the NSCL

2) The MQTT Proxy requests GSCL to build the NSCL 'uniqueContainer' Name.

3) The MQTT Proxy asks GSCL to create a < subscription > to link to the MQTT Proxy as the NSCL's "<uniqueContainer>" folder contact point data on <contentInstances>. [27]

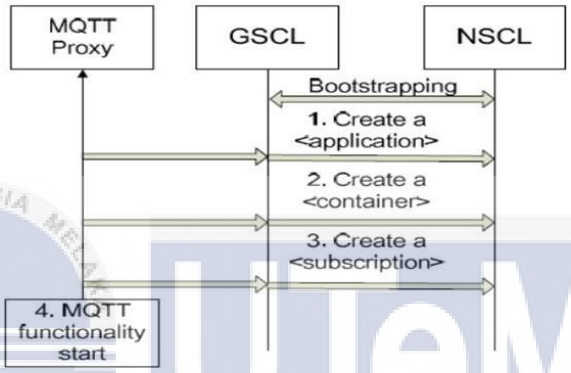


Figure 2.7 Message Flow of Part A

2.3.4 Message Flow of MQTT

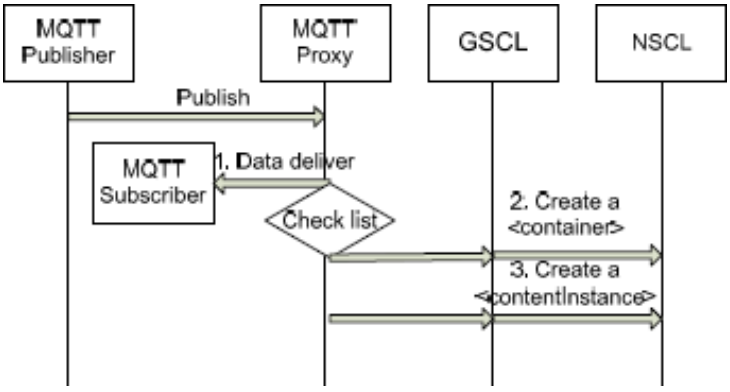


Figure 2.8 Message Flow of Part A

Fig. 2.9 depicts such a message flow. Assume that one of the MQTT clients has a MQTT packet published with the subject "ABCD":

1) This will also search to see if the topic was already registered. If yes, go to (3), else go to the next one.
2) The MQTT Proxy asks the GSCL to create an <container> with the ID “ABCD” on the NSCL.
3) The MQTT Proxy also asks the GSCL to create an <contentInstance> under the container “ABCD” on the NSCL and saves the data there

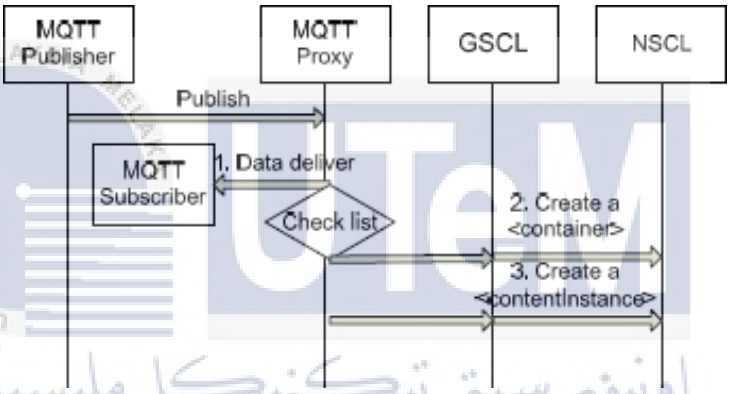


Figure 2.9 Message Flow of Part B

Next we show the distribution of OpenMTC messages to MQTT clients in Fig. 2.10 Suppose the OpenMTC "Application WXYZ" intends to publish data to MQTT customers who subscribing to the topic "WXYZ": [28]

1) The "WXYZ Application" requests that the NSCL construct a < ContentInstance > under the "UniqueContainer" container and save information about the MQTT and the subject matter there.

2) The NSCL shall use the contact point information of the MQTT Proxy to alert the MQTT Proxy through the GSCL using the < subscription > already provided during initialization.

3) The MQTT Proxy shall send the data packet to the subscribers when it receives the MQTT data and the subject informationIn addition, the MQTT Proxy may create a new vessel in the NSCL if that vessel would not exist before other NSCL applications are made available for data.

4) The MQTT Proxy requires the GSCL to construct a < container > with the NSCL "WXYZ" ID if a container does not exist in advance ..

5) The MQTT Proxy requests the GSCL to construct a <contentInstance> on the NSCL under the 'WXYZ' container and to cache the data there.

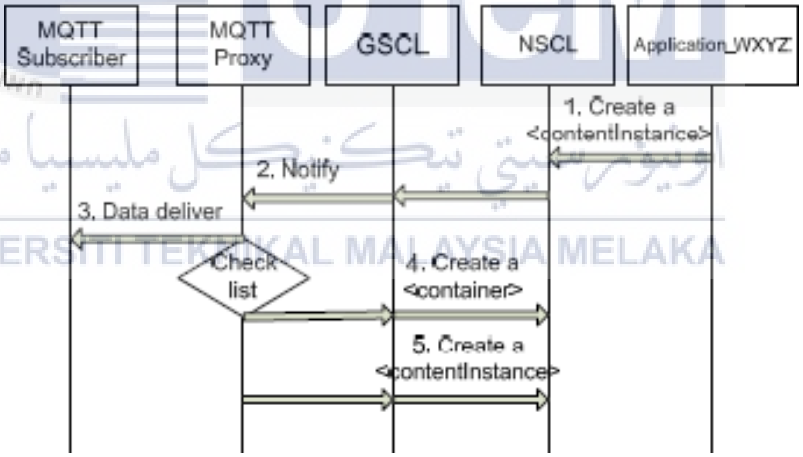


Figure 2.10 Message Flow of Part C [27]

2.4 MQTT Scope

Various applications use the MQTT in various fields. It is used in restorative administrations, Facebook notice, following and the essentialness meter, for example. In this way the MQTT show is seen as the perfect M2M and IoT contact educating show. That is an immediate consequence of its low-power controlling breaking point, lightweight , low-memory and insignificant exertion contraptions passed on in low information move limit and poor frameworks.

2.5 Quality of Service MQTT

Table 1 demonstrates three levels of quality of service (QoS) to ensure consistency of the MQTT messages. One performance is called Level 0 (at most). Messages are transmitted on a network effort basis Level 1 is (at least) one transmission. The messages are sent at least once, and can be replicated in the file. The final level is level 2 and is called one delivery (exactly). This level requires an additional protocol to ensure the message can only be sent once (i.e. the highest level of QoS). The table below provides a description of QoS levels and their definitions. [29]

Table 1 Quality of Service (QoS) Levels of MQTT

QoS level	Meaning
Level 0	A message is delivered at most once and no acknowledgement of receiving is required.
Level 1	Every message is delivered at least once and confirmation of receiving a message is required
Level 2	A four-way handshake mechanism is used exactly once for the delivery of a message

2.6 Advantages of MQTT

1) Reliability: There are some options for Quality of Service (QoS) that can be used to ensure delivery.
2) Packet agnostic: The packet can hold any type of data in the payload. The information might be either text or binary. So long as the receiver knows how to perceive it, it doesn't matter.
3) Scalability: The publish/subscribe model scales well in a power-efficient way.
4) Decoupled software: There are several development features that decouple the system and the client server, resulting in a more reliable communication strategy.
5) Time: A device can publish its data regardless of the state of the subscribing server.

2.7 Limitation of MQTT

Resource Constrained Device : There are many devices categorized as a restricted device that is further divided into three classes according to RFC 7228 based on their RAM and ROM as follows.

Table 2 Class in constrained device [30]

Class	RAM (Data Size)	Flash (Code Size)
Class 0	<<10KB	<<100KB
Class 1	~10kb	~100KB
Class 2	~50KB	~250KB

Due to the very limited computing performance, most resource-constrained devices, especially class 0 devices, are unable to handle most security approaches, particularly the mechanism with heavy computing such as running TLS for transport security.

1) Tremendous number of gadgets: The critical number of associated gadgets will in general produce extra vulnerabilities. It is repetitive for the IT office to deal with a few distinct kinds of gadgets, especially when the assurance component applies to the IoT framework. For instance, the IT office should put forth a great deal of attempt to keep the security certifications confirmed by utilizing the username and secret key.

2) Lack of security awareness: Lack of security mindfulness permits designers to pick usefulness over wellbeing when exchange offs are required. Bitdefender study concentrate in the U.S., Romania, Germany , Australia, France, and UK, just under half of people in every nation who know about practically all boundaries of security mindfulness (for example protection concerns, loss of shrewd gadget control, programming update recurrence). The HP Fortify report noticed that 70 percent of clients are utilizing decoded arrange administration. [31]

2.8 Conclusion

Based on background studies about MQTT above, I have choose MQQT Protocol as my main protocol in doing this project because of its advantages in low cost and its reliable. This two factors is important to make sure there is no issue or complication when applying this project.

CHAPTER 3

METHODOLOGY

3.0 Project Management

The purpose of project management is to ensure that the project task completely meet the requirements and objectives of projects by applying the knowledge, skills and techniques that we achieve. The chapter defines the scope and limitations of the investigations and situates the investigation amongst other existing project.

3.1 Flow Chart

This section describes the methodology used during implements this project. The step throughout the project will be explained as below and referring to the Figure 3.2.

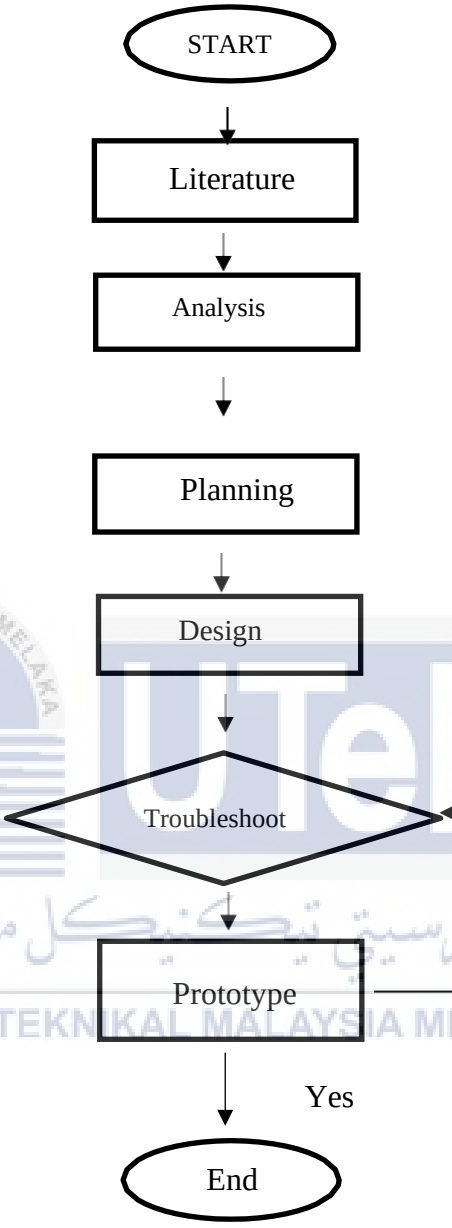


Figure 3.1 Flow Chart of the Methodology

Description of each stage of the flow chart :

I. START :

- Start the project after get approval from supervisor and panel

II. LITERATURE REVIEW :

- Read the previous research and do the literature review on current method and technology.
- Identify the limitation for MQTT protocol.

III. PLANNING :

- Planning the further step such as which software and hardware that needed in doing this project.

IV. DESIGN :

- Design the circuit diagram for this project by using the hardware and software part.

V. TROUBLESHOOT :

- Fix the problem that have been faced when doing the project circuit and project coding

VI. PROTOTYPE TEST :

- Test the prototype. This step is to make sure the prototype functioning based on expectation

VII. END :

- The prototype are ready to be presented and implemented at the expected place.

3.2 Block Diagram of Project Outcome

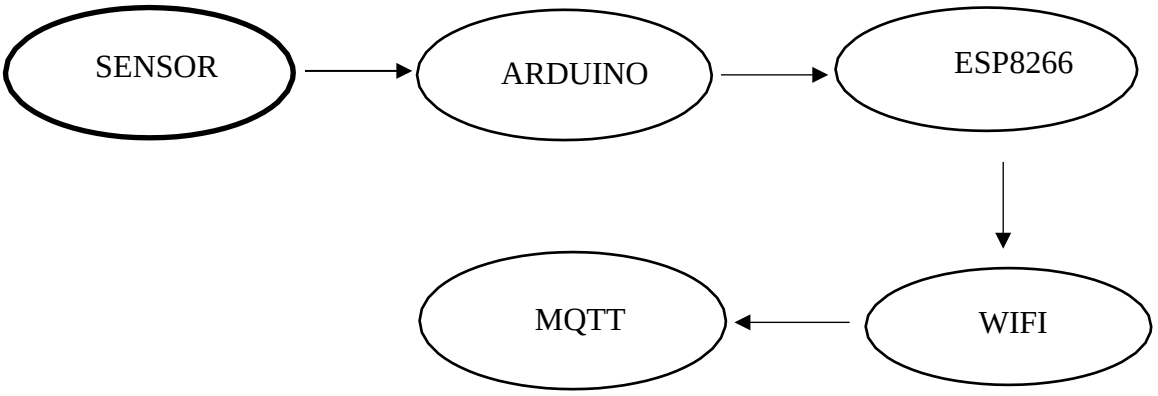


Figure 3.2 Block Diagram of Project Outcome

Description for each block diagram

I. Sensor, Buzzer :

- Circuit installation of the project by using sensor and buzzer .

II. Arduino :

- Coding implementation by using Arduino based on coding sensor and buzzer.

III. ESP8266 :

- Arduino activate the buzzer and sent data of sensor to the esp8266.

IV. WiFi SSID :

- ESP8266 will connect to the wifi to send the data

V. MQTT Protocol :

- Data sent at ESP8266 will transferred to the MQTT Protocol

This project is divided into two part that is circuit implementation and programming part. Below are the explanation about each of the part. At software part, there explanation about software used in this project while at hardware part about the component used in this project.

3.3 Software Part

Fritzing

Fritzing is an initiative in the field of open source hardware that makes electronics accessible to anyone as creative material. In the spirit of Processing and Arduino, we offer a software tool, a community website and services that fosters a creative ecosystem that allows users to document their prototypes, share with others, teach electronics in a classroom, and layout and manufacture professional pcbs. Fritzing comes with tons of already-installed electronic parts with the software. So this Fritzing software had been chosen to make this project's circuit operation..

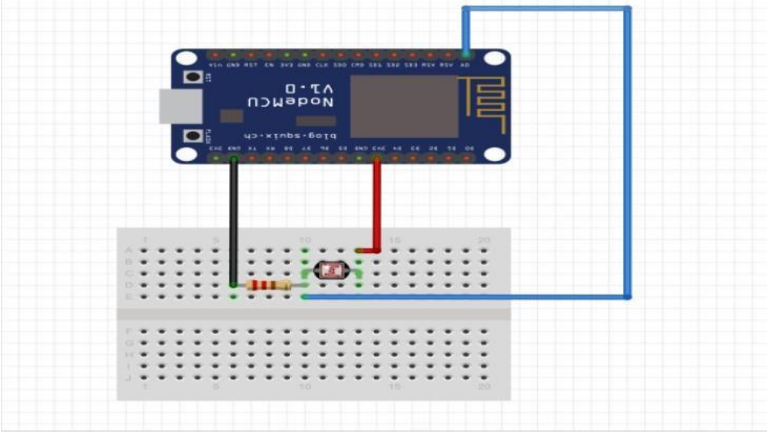


Figure 3.3 Proposed circuit diagram

Arduino

Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its products are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form or as do-it-yourself (DIY) kits..

LightSensor | Arduino 1.8.10

File Edit Sketch Tools Help

LightSensor

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
//
#define lightsensor A0

// Update these with values suitable for your network.
const char* ssid = "R4J@unifi";//put your hotspot ssid here
const char* password = "rf1998akmal95";//put your hotspot password here
const char* mqtt_server = "broker.mqtt-dashboard.com";//put your mqtt server address/url here
//const char* mqtt_server = "iot.eclipse.org";

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;
int threshold=50; // you might need to adjust this value to define light on/off status
void setup_wifi() {
   delay(100);
  // We start by connecting to a WiFi network
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED)
    {
      delay(500);
      Serial.print(".");
    }
  randomSeed(micros());
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

Upload

LightSensor

```
}

void callback(char* topic, byte* payload, unsigned int length)
{
} //end callback

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected())
  {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    //if you MQTT broker has clientID,username and password
    //please change following line to    if (client.connect(clientId,userName,passWord))
    if (client.connect(clientId.c_str()))
    {
      Serial.println("connected");
     //once connected to MQTT broker, subscribe command if any
      client.subscribe("OsoyooCommand");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 6 seconds before retrying
      delay(6000);
    }
  }
} //end reconnect()

void setup() {
  Serial.begin(115200);
  setup_wifi();
```

```
void loop()  {
  if (!client.connected())  {
    reconnect();
  }
  client.loop();
  long now = millis();
//send data every second
  if (now - lastMsg > 500)  {
      lastMsg = now;
     int val=analogRead(lightsensor);
      String msg="real time light strength: ";
      msg= msg+ val;
    if (val>threhold)
       msg="0:  "+msg;
     else
       msg="1:  "+msg;
      char message[58];
      msg.toCharArray(message,58);
      Serial.println(message);

      //publish sensor data to MQTT broker

       client.publish("LightData", message);
  }
}
```

Figure 3.4 Coding for Reading Light Sensor Data

LightSensor | Arduino 1.8.10

File Edit Sketch Tools Help

LightSensor

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
//
#define lightsensor A0

// Update these with values suitable for your network.
const char* ssid = "R4J@unifi";//put your hotspot ssid here
const char* password = "rf1998akmal95";//put your hotspot password here
const char* mqtt_server = "broker.mqtt-dashboard.com";//put your mqtt server address/url here
//const char* mqtt_server = "iot.eclipse.org";
```

Figure 3.5 Coding for connect to the MQTT server

MQTT Box

MQTTBox is a helper program to develop and load test MQTT based clients, brokers, devices, cloud and apps. Every aspect of MQTTBox is specifically designed to maximize development and testing productivity. Together with integrated MQTT clients and load testing tools, its efficient enough to overwhelm the MQTT workflow.

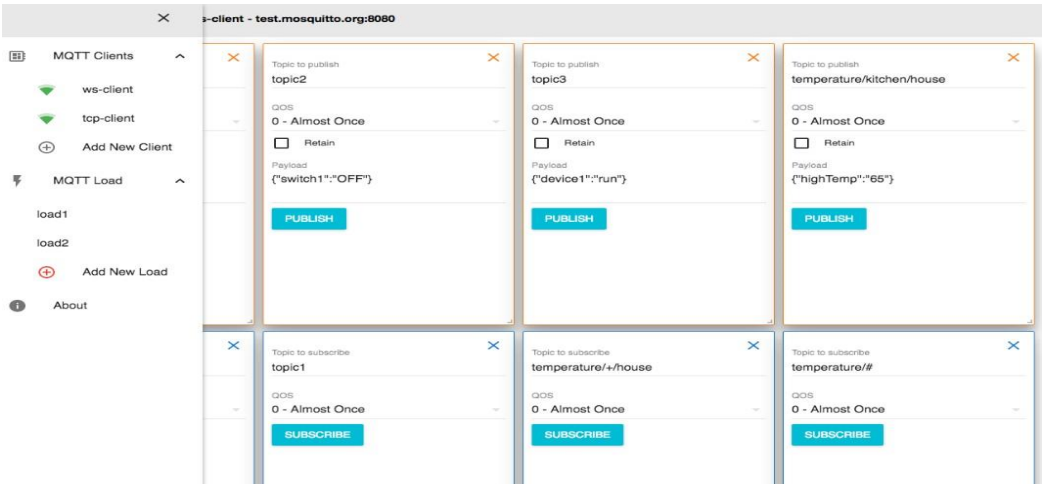


Figure 3.6 Navigation between MQTT clients and MQTT Load test cases.

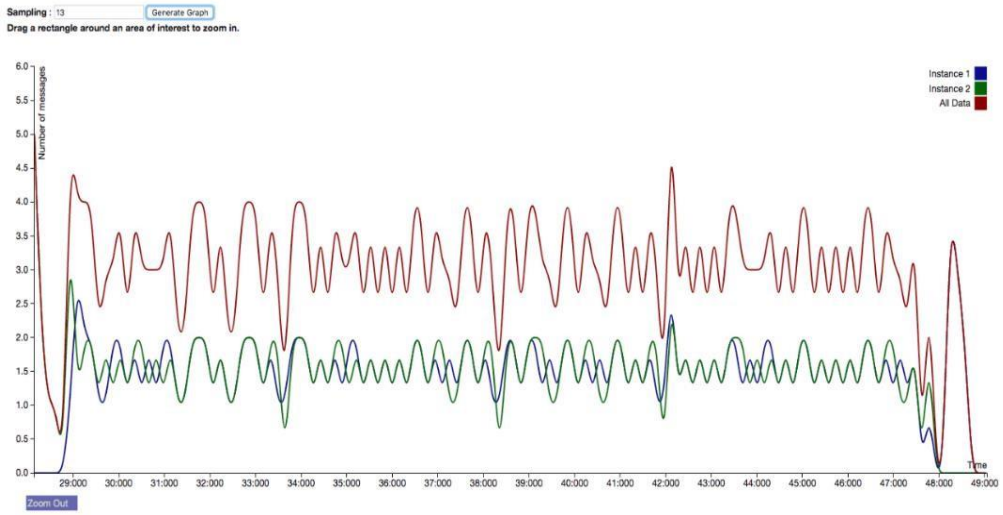


Figure 3.7 Graph format plotted with number of messages vs time

3.4 Hardware Part

NodeMCU

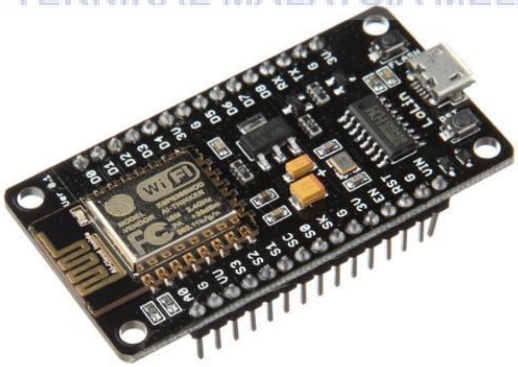


Figure 3.8 NodeMCU ESP8266V3

Resistor 1K Ohm

Figure 3.9 Resistor

LDR

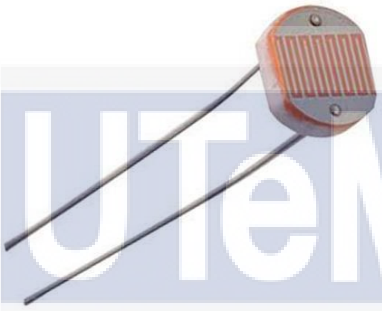


Figure 3.10 LDR

3.5 Conclusion

The flow of the project done according to the flowchart stated. Both the circuit implementation and software part were built up based on schematic circuits. The results of the project for each parts were observed and recorded thus further analyzed and discussed on Chapter 4.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Connection for Project

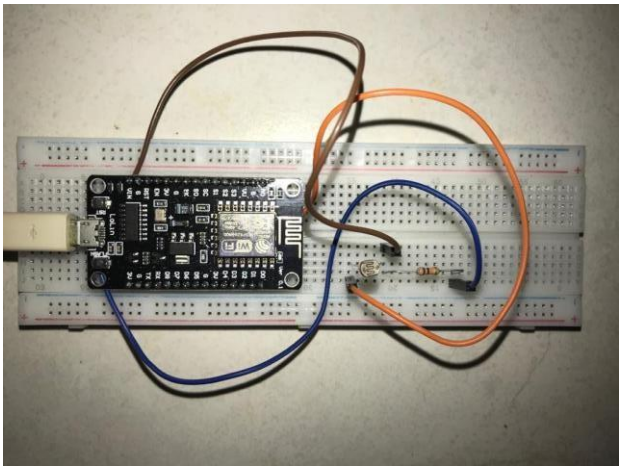


Figure 4.1 Connection of circuit

4.2 MQTT Client

Figure 4.2 shows that the difference when the server connected or not. When its connected to the server, it will show Connected and the border is in green colour. When the connection cannot connect to the server, it will show Connection Error with red border around its function. The connection connected or error depends on the settings inside the function the variable at settings need to correct with the server.

Figure 4.2 Difference between connection

Figure 4.3 show the variable need to set correctly in the setting for each function. Variable need to set correctly if want to connect to the server.

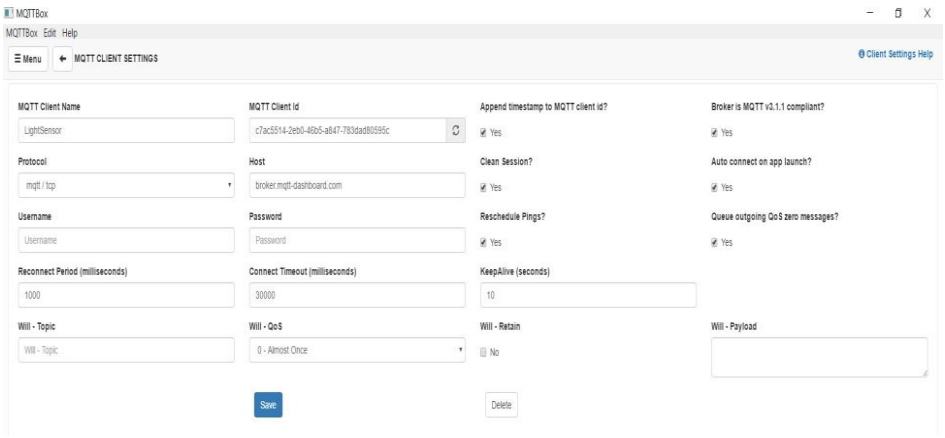


Figure 4.3 The variable in setting.

Figure 4.4 show topic that can be publish or topic to be subscribe. Noted that this is the subscriber, so function that use here is topic to subscribe. When the topic to subscribe that is LightData is correct with the server, the information from server received at the function topic to subscribe.

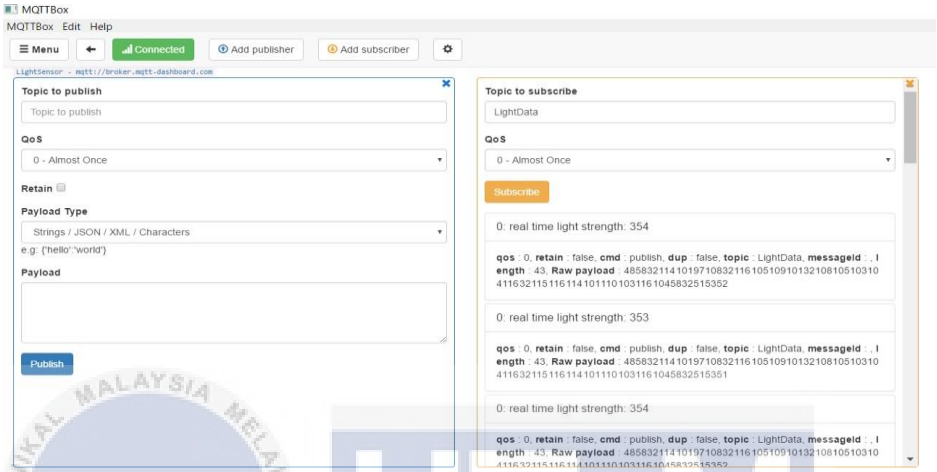


Figure 4.4 Function to topic

Figure 4.5 show that display at Arduino that act as server. The light data that taken at server will be displayed at serial monitor and the the subscriber will received the same data if subscribed at correct topic.

Figure 4.5 Display at serial monitor

4.3 MQTT Load Part 1

Figure 4.6 show that data taken when button Start Load Test were pressed. At instance 1, the time taken for subscribed is 9.8500s and the message received is set to 20 message. At instance 2, the time taken for 20 message to be subscribed is 9.8510s.

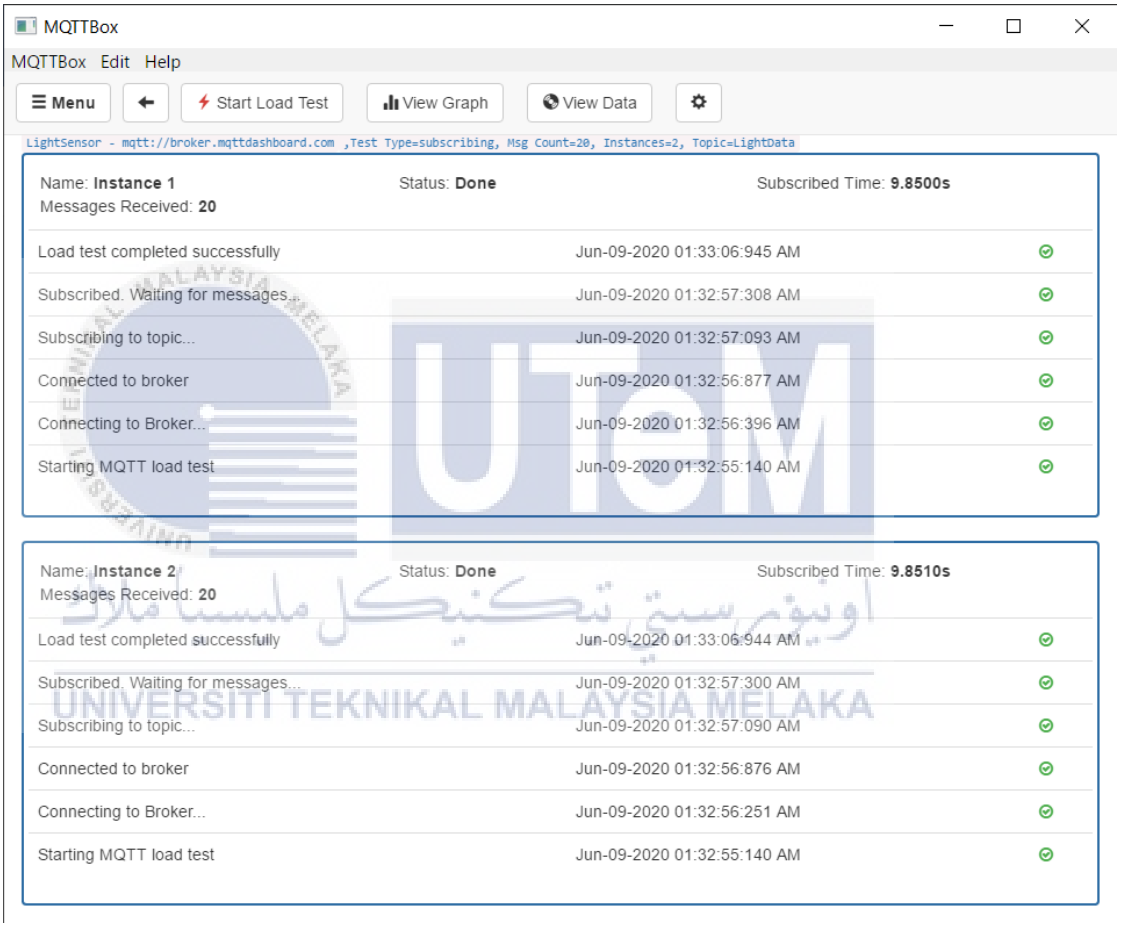


Figure 4.6 Data taken for subscribed message

Figure 4.7 show data taken for instance 1 within subscribed time.

MQTTBox

MQTTBox Edit Help

Menu | Instance 1

Time	Message Id	Topic	QoS	Instance	Payload
Jun-09-2020 01:32:57:364 AM		LightData	0	1	0: real time light strength: 343
Jun-09-2020 01:32:57:900 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:32:58:400 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:32:58:870 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:32:59:394 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:32:59:868 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:33:00:390 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:33:00:868 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:33:01:430 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:33:01:910 AM		LightData	0	1	0: real time light strength: 341
Jun-09-2020 01:33:02:389 AM		LightData	0	1	0: real time light strength: 343
Jun-09-2020 01:33:02:923 AM		LightData	0	1	0: real time light strength: 344
Jun-09-2020 01:33:03:396 AM		LightData	0	1	0: real time light strength: 344
Jun-09-2020 01:33:03:992 AM		LightData	0	1	0: real time light strength: 343
Jun-09-2020 01:33:04:401 AM		LightData	0	1	0: real time light strength: 343
Jun-09-2020 01:33:04:952 AM		LightData	0	1	0: real time light strength: 343
Jun-09-2020 01:33:05:433 AM		LightData	0	1	0: real time light strength: 343
Jun-09-2020 01:33:05:910 AM		LightData	0	1	0: real time light strength: 336
Jun-09-2020 01:33:06:430 AM		LightData	0	1	0: real time light strength: 337
Jun-09-2020 01:33:06:943 AM		LightData	0	1	0: real time light strength: 339

Figure 4.7 Data for instance 1

Figure 4.8 show data taken for instance 2 within subscribed time.

MQTTBox

MQTTBox Edit Help

Menu | Instance 2

Time	Message Id	Topic	QoS	Instance	Payload
Jun-09-2020 01:32:57:361 AM		LightData	0	2	0: real time light strength: 343
Jun-09-2020 01:32:57:901 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:32:58:398 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:32:58:870 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:32:59:391 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:32:59:865 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:33:00:391 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:33:00:871 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:33:01:432 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:33:01:909 AM		LightData	0	2	0: real time light strength: 341
Jun-09-2020 01:33:02:392 AM		LightData	0	2	0: real time light strength: 343
Jun-09-2020 01:33:02:927 AM		LightData	0	2	0: real time light strength: 344
Jun-09-2020 01:33:03:396 AM		LightData	0	2	0: real time light strength: 344
Jun-09-2020 01:33:03:993 AM		LightData	0	2	0: real time light strength: 343
Jun-09-2020 01:33:04:398 AM		LightData	0	2	0: real time light strength: 343
Jun-09-2020 01:33:04:951 AM		LightData	0	2	0: real time light strength: 343
Jun-09-2020 01:33:05:432 AM		LightData	0	2	0: real time light strength: 343
Jun-09-2020 01:33:05:900 AM		LightData	0	2	0: real time light strength: 336
Jun-09-2020 01:33:06:422 AM		LightData	0	2	0: real time light strength: 337
Jun-09-2020 01:33:06:941 AM		LightData	0	2	0: real time light strength: 339

Figure 4.8 Data for instance 2

Figure 4.9 show graph format plotted with number of messages vs time Instance 1 for 5 sampling. Total time taken for 5 sampling is 9.575s. Time interval for each plot is 1.915.

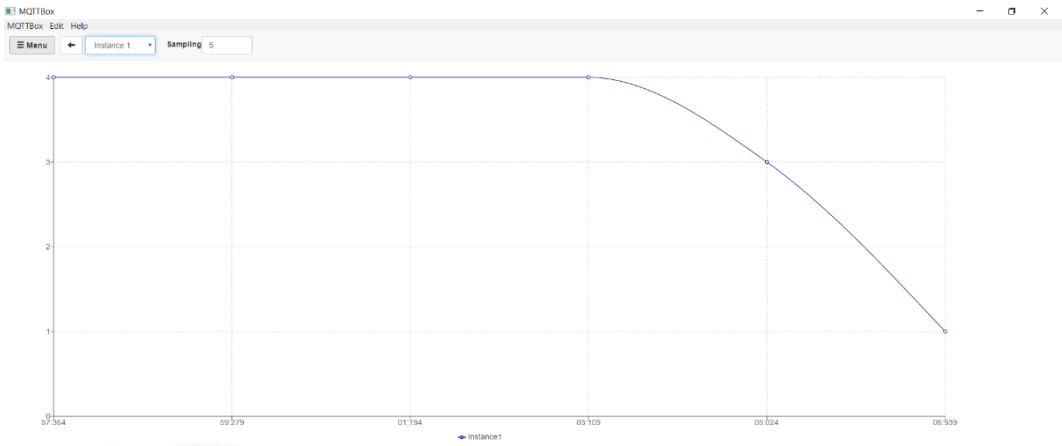


Figure 4.9 Instance 1 for 5 sampling

Figure 4.10 show graph format plotted with number of messages vs time Instance 1 for 10 sampling. Total time taken for 10 sampling is 9.575s. Time interval for each plot is 0.957s.

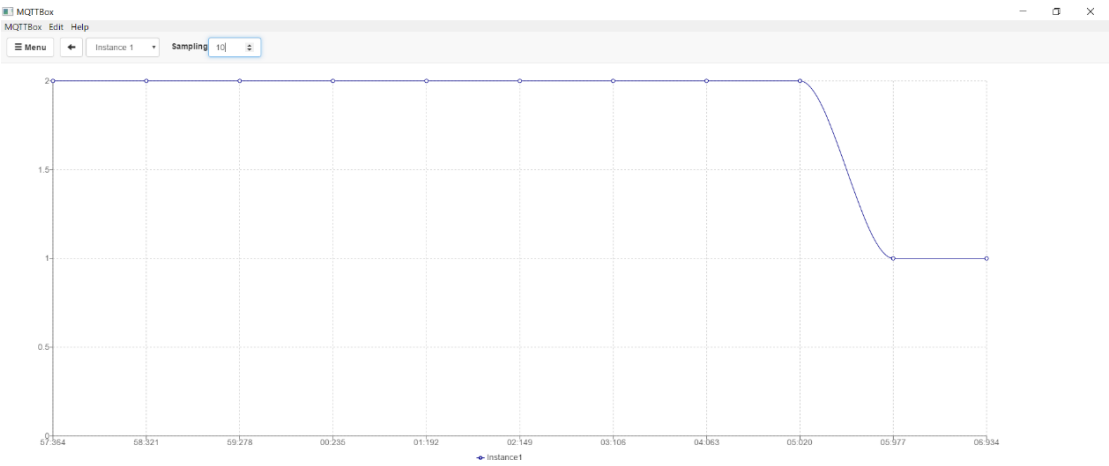


Figure 4.10 Instance 1 for 10 sampling

Figure 4.11 show graph format plotted with number of messages vs time Instance 1 for 15 sampling. Total time taken for 15 sampling is 9.575s. Time interval for each plot is 0.638s.

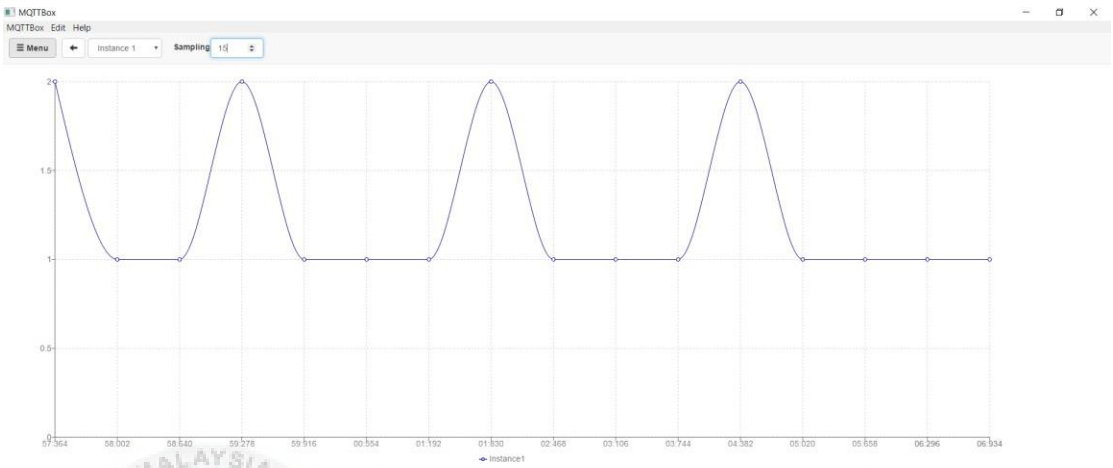


Figure 4.11 Instance 1 for 15 sampling

Figure 4.12 show graph format plotted with number of messages vs time Instance 2 for 5 sampling. Total time taken for 5 sampling is 9.580s. Time interval for each plot is 1.916s.

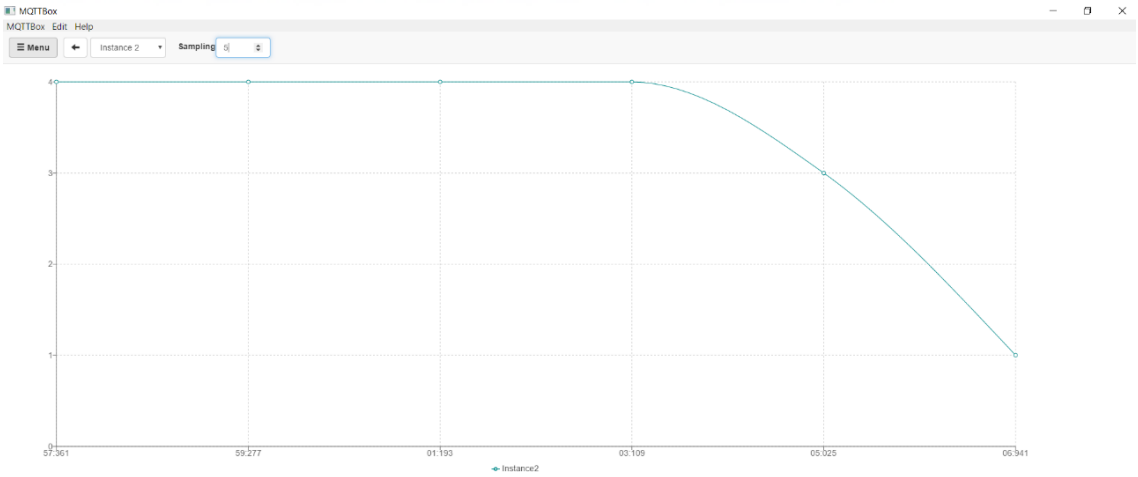


Figure 4.12 Instance 2 for 5 sampling

Figure 4.13 show graph format plotted with number of messages vs time Instance 2 for 10 sampling. Total time taken for 10 sampling is 9.580s. Time interval for each plot is 0.958s.

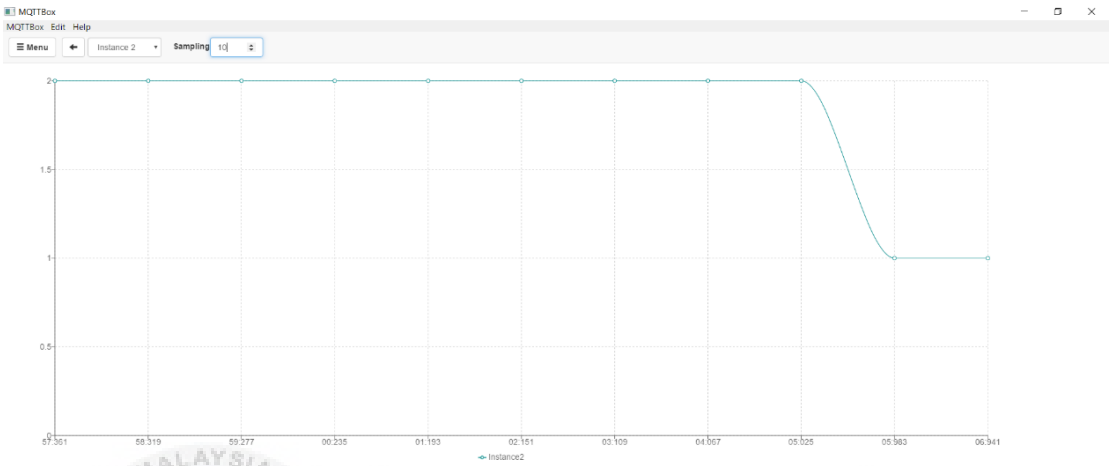


Figure 4.13 Instance 2 for 10 sampling

Figure 4.14 show graph format plotted with number of messages vs time Instance 2 for 15 sampling. Total time taken for 15 sampling is 9.580s. Time interval for each plot is 0.639s.

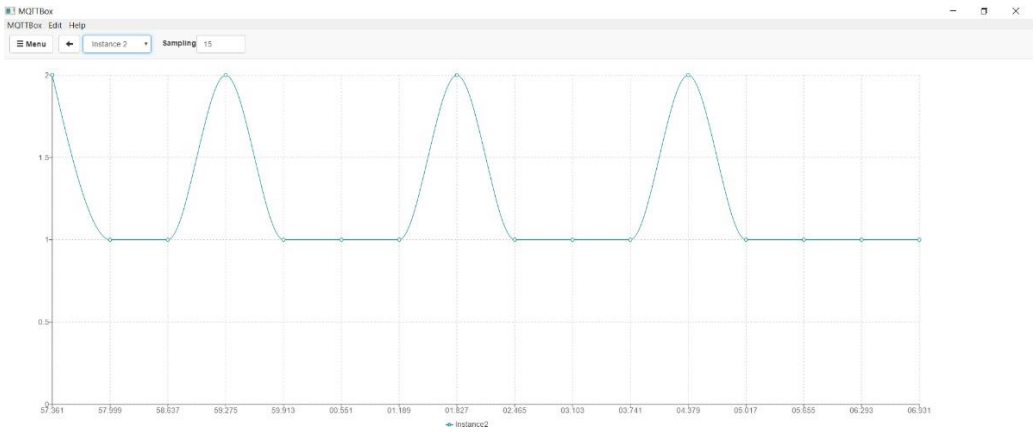


Figure 4.14 Instance 2 for 15 sampling

Figure 4.15 show graph format plotted with number of messages vs time all instance for 5 sampling. Total time taken for 5 sampling is 9.580s. Time interval for each plot is 1.916s.

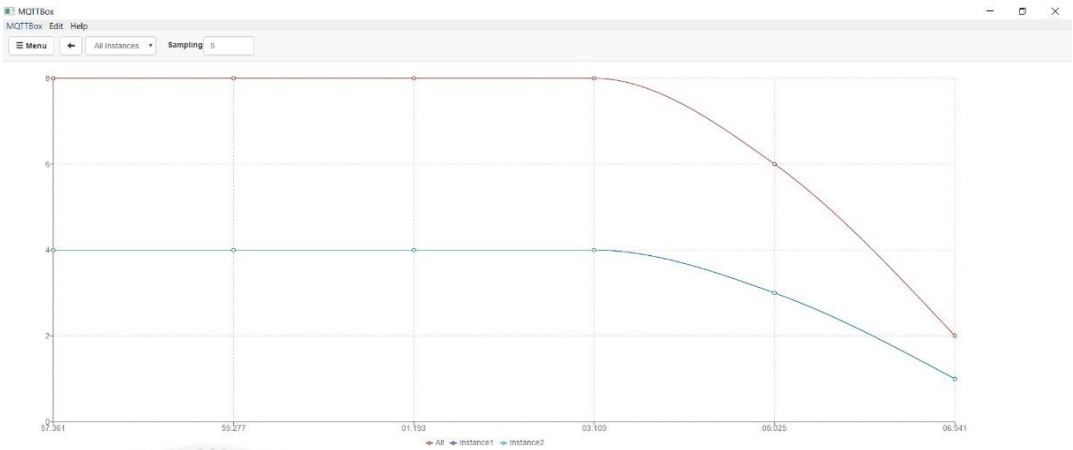


Figure 4.15 All instance for 5 sampling

Figure 4.16 show that show graph format plotted with number of messages vs time all instance for 10 sampling. Total time taken for 10 sampling is 9.580s. Time interval for each plot is 0.958s.

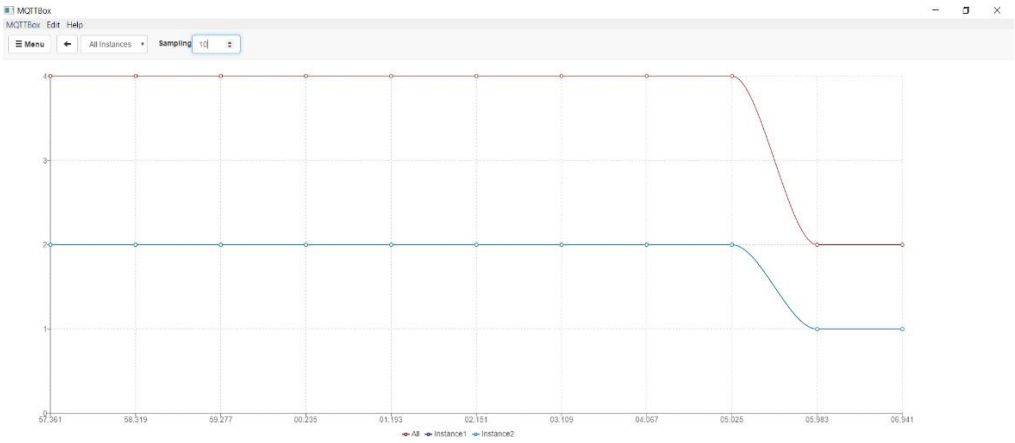


Figure 4.16 All instance for 10 sampling

Figure 4.17 show graph format plotted with number of messages vs time all instance for 15 sampling. Total time taken for 15 sampling is 9.580s. Time interval for each plot is 0.639s.

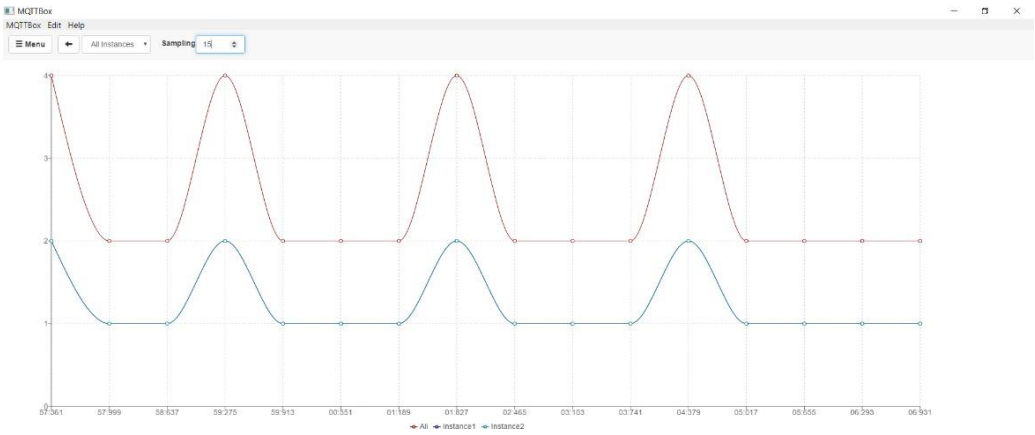


Figure 4.17 All instance for 15 sampling

4.4 Discussion

Based on the instance 1 graph that shown at this part, the total time taken for all sampling for instance 1 is 9.575s while at the data for subscribed message is 9.850s. So delay detected in this part is 0.275s.

Based on the instance 2, total time taken for all sampling for instance 2 is 9.580s while at data for subscribed message is 9.8510s. So delay at part instance 2 is 0.271s.

This can be concluded that the data received from server also having delay issue although the delay below 1s.

4.5 MQTT Load Part 2

Figure 4.18 show that data taken when button Start Load Test were pressed. At instance 1, the time taken for subscribed is 9.9780s and the message received is set to 20 message. At instance 2, the time taken for 20 message to be subscribe is 10.0200s.

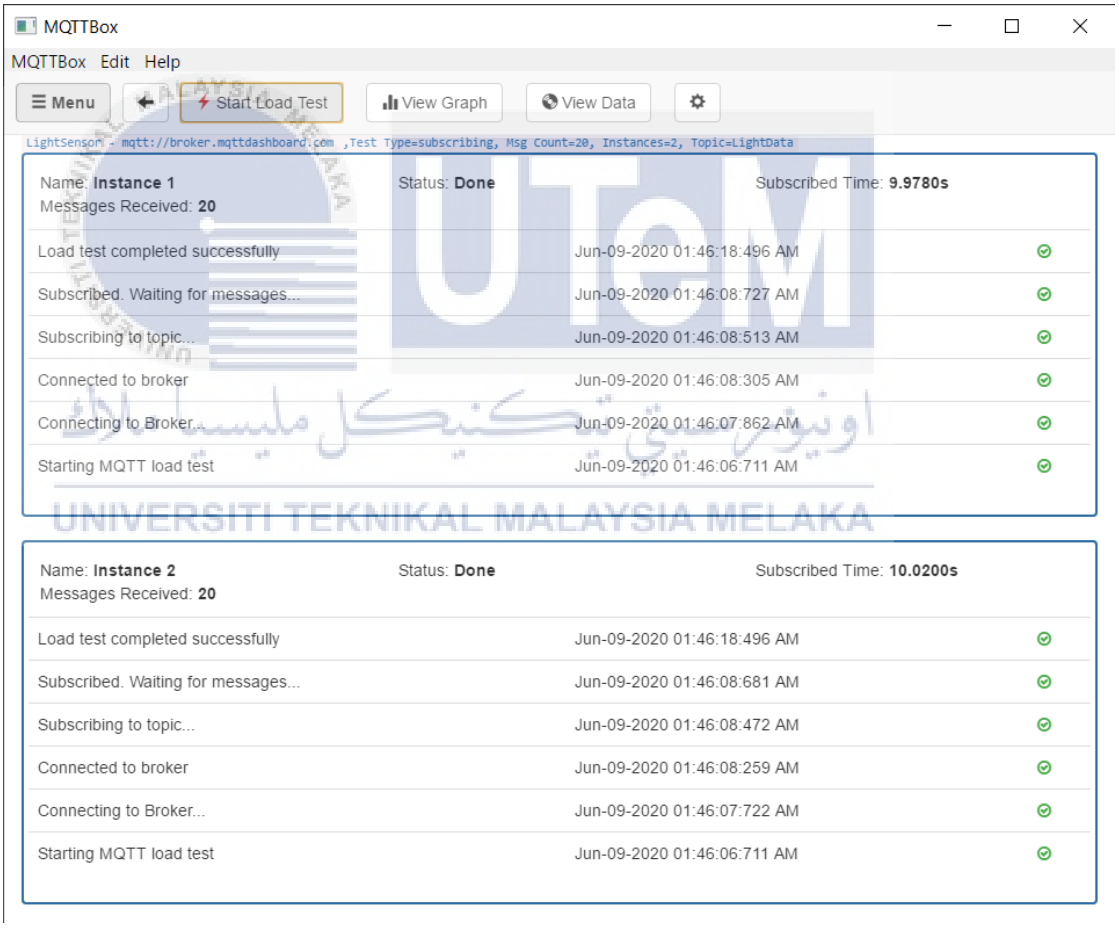


Figure 4.18 Data taken for subscribed message

Figure 4.19 show data taken for instance 1 within subscribed time.

MQTTBox

MQTTBox Edit Help

Menu | Instance 1

Time	Message Id	Topic	QoS	Instance	Payload
Jun-09-2020 01:46:08:954 AM		LightData	0	1	0: real time light strength: 356
Jun-09-2020 01:46:09:472 AM		LightData	0	1	0: real time light strength: 356
Jun-09-2020 01:46:09:942 AM		LightData	0	1	0: real time light strength: 356
Jun-09-2020 01:46:10:438 AM		LightData	0	1	0: real time light strength: 356
Jun-09-2020 01:46:10:935 AM		LightData	0	1	0: real time light strength: 356
Jun-09-2020 01:46:11:462 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:11:949 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:12:461 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:12:960 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:13:463 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:13:962 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:14:474 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:14:981 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:15:470 AM		LightData	0	1	0: real time light strength: 356
Jun-09-2020 01:46:15:949 AM		LightData	0	1	0: real time light strength: 356
Jun-09-2020 01:46:16:510 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:16:979 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:17:478 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:17:972 AM		LightData	0	1	0: real time light strength: 355
Jun-09-2020 01:46:18:491 AM		LightData	0	1	0: real time light strength: 355

Figure 4.19 Data for instance 1

Figure 4.20 show data taken for instance 2 within subscribed time.

MQTTBox

MQTTBox Edit Help

Menu | Instance 2

Time	Message Id	Topic	QoS	Instance	Payload
Jun-09-2020 01:46:08:953 AM		LightData	0	2	0: real time light strength: 356
Jun-09-2020 01:46:09:472 AM		LightData	0	2	0: real time light strength: 356
Jun-09-2020 01:46:09:943 AM		LightData	0	2	0: real time light strength: 356
Jun-09-2020 01:46:10:439 AM		LightData	0	2	0: real time light strength: 356
Jun-09-2020 01:46:10:936 AM		LightData	0	2	0: real time light strength: 356
Jun-09-2020 01:46:11:462 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:11:949 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:12:461 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:12:964 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:13:462 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:13:962 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:14:470 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:14:983 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:15:470 AM		LightData	0	2	0: real time light strength: 356
Jun-09-2020 01:46:15:952 AM		LightData	0	2	0: real time light strength: 356
Jun-09-2020 01:46:16:509 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:16:981 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:17:484 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:17:972 AM		LightData	0	2	0: real time light strength: 355
Jun-09-2020 01:46:18:492 AM		LightData	0	2	0: real time light strength: 355

Figure 4.20 Data for instance 2

Figure 4.21 show graph format plotted with number of messages vs time Instance 1 for 5 sampling. Total time taken for 5 sampling is 9.535s. Time interval for each plot is 1.907s.

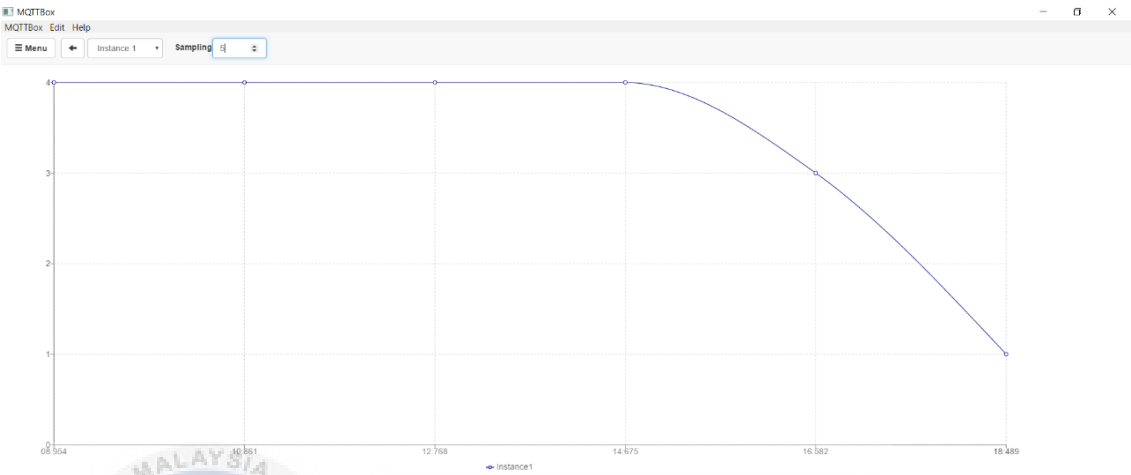


Figure 4.21 Instance 1 for 5 sampling

Figure 4.22 show graph format plotted with number of messages vs time Instance 1 for 10 sampling. Total time taken for 10 sampling is 9.535s. Time interval for each plot is 0.935s.

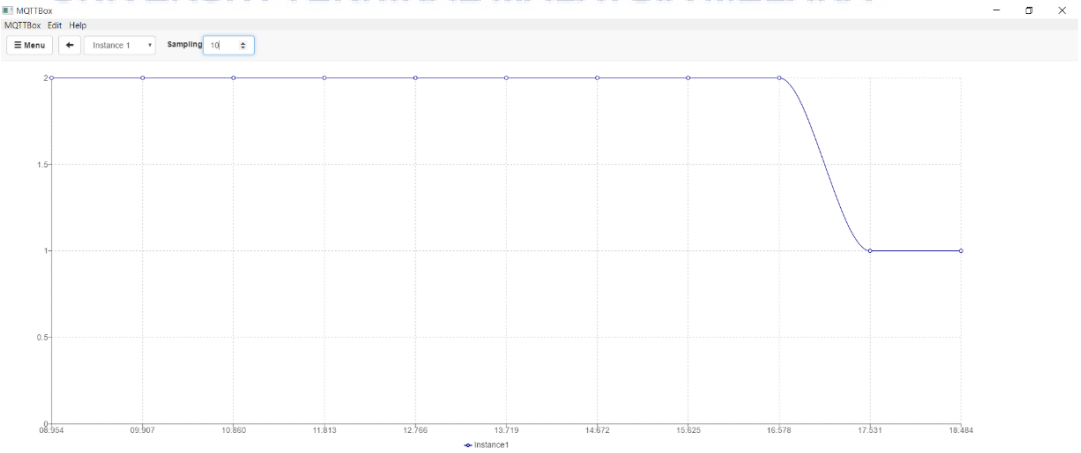


Figure 4.22 Instance 1 for 10 sampling

Figure 4.23 show graph format plotted with number of messages vs time Instance 1 for 15 sampling. Total time taken for 15 sampling is 9.535s. Time interval for each plot is 0.636s.

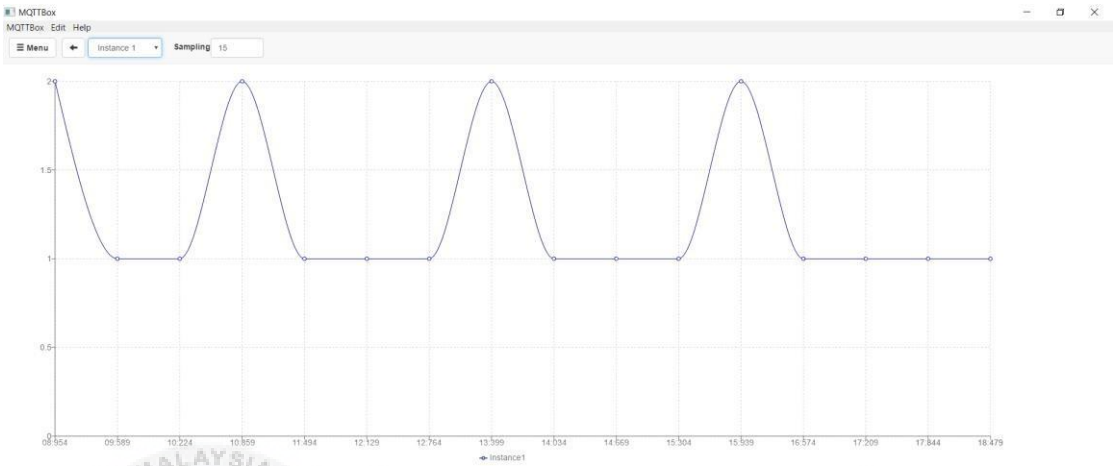


Figure 4.23 Instance 1 for 15 sampling

Figure 4.24 show graph format plotted with number of messages vs time Instance 2 for 5 sampling. Total time taken for 5 sampling is 9.535s. Time interval for each plot is 1.907s.

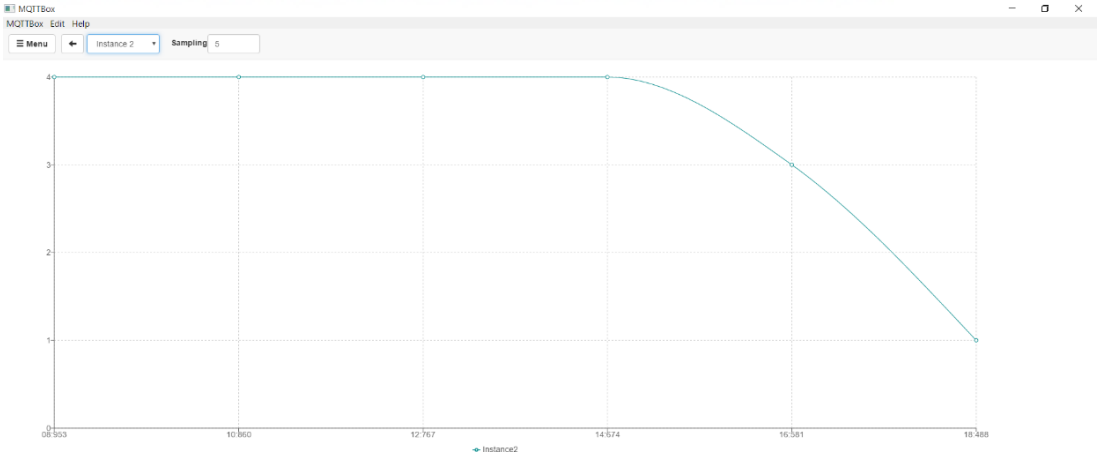


Figure 4.24 Instance 2 for 5 sampling

Figure 4.25 show graph format plotted with number of messages vs time Instance 1 for 5 sampling. Total time taken for 5 sampling is 9.535s. Time interval for each plot is 0.935s.

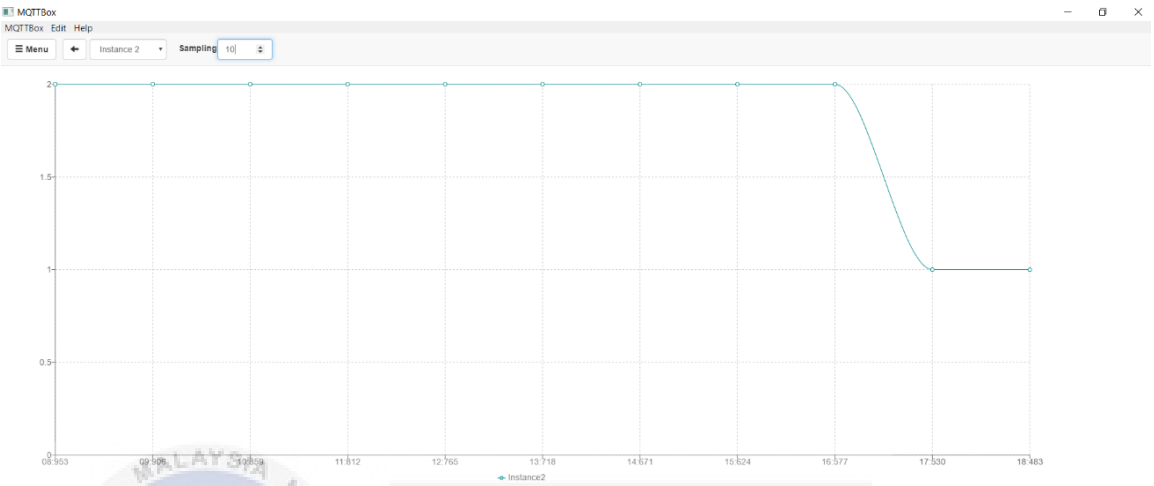


Figure 4.25 Instance 2 for 10 sampling

Figure 4.26 show graph format plotted with number of messages vs time Instance 1 for 15 sampling. Total time taken for 15 sampling is 9.535s. Time interval for each plot is 0.636s.

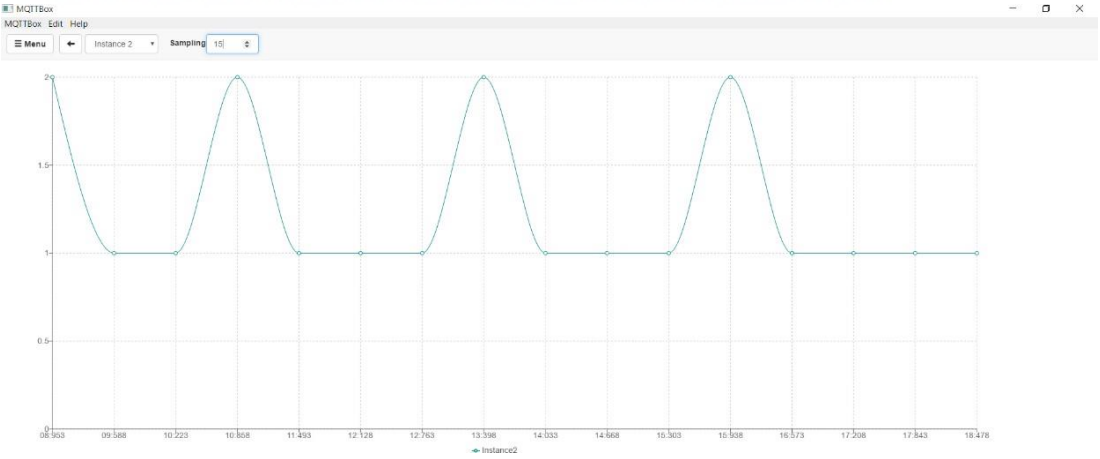


Figure 4.26 Instance 2 for 15 sampling

Figure 4.27 show graph format plotted with number of messages vs time all instance for 5 sampling. Total time taken for 5 sampling is 9.535s. Time interval for each plot is 1.907s.

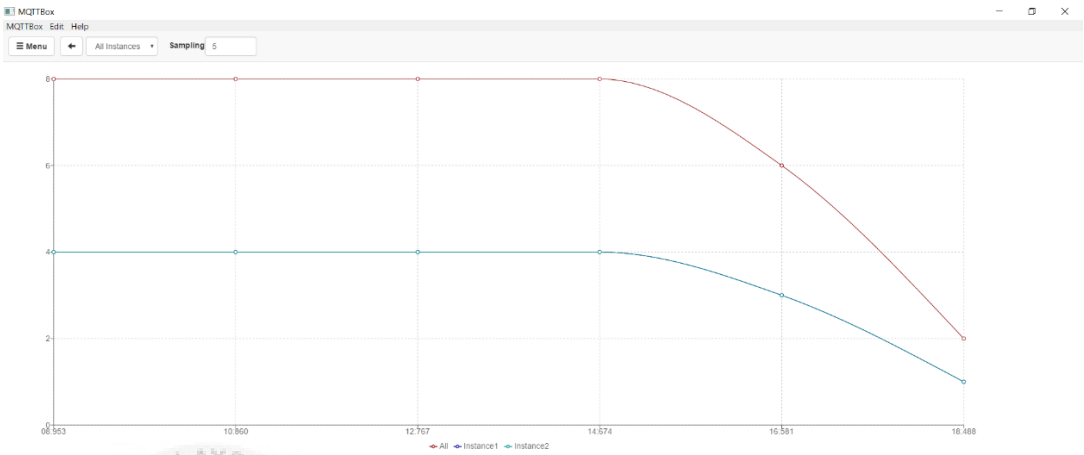


Figure 4.27 All instance for 5 sampling

Figure 4.28 show graph format plotted with number of messages vs time all instance for 10 sampling. Total time taken for 10 sampling is 9.535s. Time interval for each plot is 0.935s.

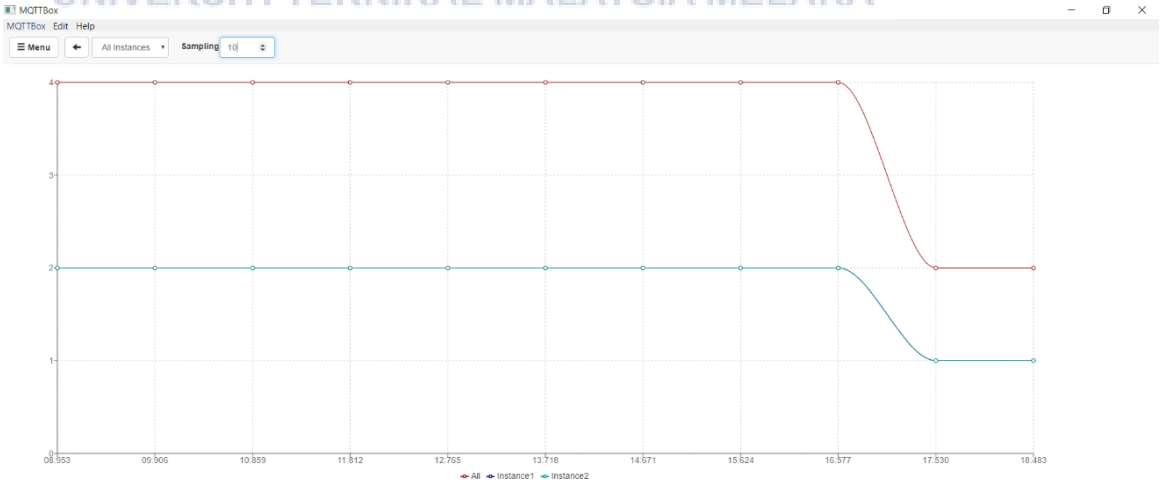


Figure 4.28 All instance for 10 sampling

Figure 4.29 show graph format plotted with number of messages vs time all instance for 15 sampling. Total time taken for 15 sampling is 9.535s. Time interval for each plot is 0.636s.

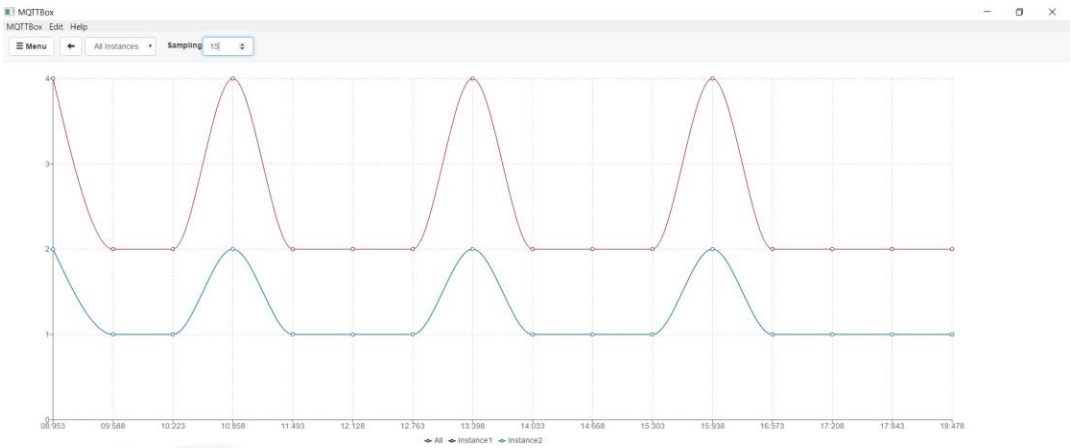


Figure 4.29 All instance for 15 sampling

4.6 Discussion

Based on the instance 1 graph that shown at this part, the total time taken for all sampling for instance 1 is 9.535s while at the data for subscribed message is 9.9780s. So delay detected in this part is 0.443s.

Based on the instance 2, total time taken for all sampling for instance 2 is also 9.535s while at data for subscribed message is 10.020s. So delay at part instance 2 is 0.485s.

Although there are delay in receiving data from subscriber, the delay is below 1s, so this can conclude that the MQTT protocol is consistent . Its also shows that accurate data that received by the receiver.

CHAPTER 5

CONCLUSION AND FUTURE WORKS

A summary of entire study has been constructed. It too includes the key offers the thesis has made the current state of knowledge and exercise. It also plot the guidelines for upcoming exploration by suggesting ideas grounded on the existing work.

5.1 Conclusion

The research influence of this thesis can be brief as follows :

- MQTT protocol is suitable for communication because although it facing delay issue, the delay is not too far from the actual time taken for data received that is below 1s.
- This thesis also shows that MQTT protocol is a suitable platform for sending and receiving data because the its accuracy for sending data without any loss.

5.2 Future Works

The work shown in this thesis is majority contructed on simulation. Only one software that is MQTTBox, so the outcomes are yet to be verified in an extensive situation. In future, we aim to try this framework by using another software which can prove this protocol is the best protocol for performaces of notification alert system.

REFERENCES

[1] Steven E. Kozisek, Carl M. Coppage,Rober J. Morrill, "SYSTEM AND METHOD FOR MONITORING AND OPTIMIZING NETWORK PERFORMANCE TO A WIRELESS DEVICE," p. 1, May 31, 2007.

[2] Patricio Humberto Saavedra,, "SYSTEM, APPARATUS AND METHOD FOR PROVIDING IMPROVED PERFORMANCE OF AGGREGATED/BONDED NETWORK CONNECTIONS WITH CLOUD PROVISIONING," p. 1, Mar. 4, 2015.

[3] Beisel, Philipp W, "POLLING-BASED SECURE NETWORK MESSAGE NOTIFICATION SYSTEM AND METHOD WITH PERFORMANCE ENHANCING FEATURES," p. 1, Jul. 1, 2011.

[4] Michael Martin Saigh, Kevin Richard Arndt, Andrew Victor Saigh, "PERSONAL SAFETY MOBILE NOTIFICATION SYSTEM," p. 1, Jul. 20, 2012.

[5] John M. Suit. Mark J., "MONITORING SYSTEM PERFORMANCE CHANGES BASED ON CONFIGURATION MODIFICATION," p. 1, Oct. 15, 2010.

[6] Philip A. Reitmour, Benjamin F. Reitmour, Nicholas R. Reitmour, John T. Reitmour, "NOTIFICATION AND TRACKING SYSTEM FORMOBILE DEVICES," p. 1, Sep. 5, 2012.

[7] K. Govindan and A. P. Azad,, "End-to-end service assurance in loT MQTT-SN," *2015 12th Annual IEEE Consumer Communications and Networking Conference,*, pp. 290-296, 2015..

[8] Riccardo Giambona, Alessandro E. C. Redondi, Matteo Cesana, "MQTT+: Enhanced Syntax and Broker Functionalities for Data Filtering, Processing and Aggregation," p. 77, 2018.

[9] Grgi, Krešimir, Ivan Špeh, and Ivan Hei., ""A web-based IoT solution for monitoring data using MQTT protocol.","*Smart Systems and Technologies (SST), International Conference* , p. 1, 2016.

[10] Shubhangi A. Shinde, Pooja A. Nimkar, Shubhangi P. Singh, Vrushali D. Salpe, Yogesh R. Jadhav, "“MQTT-Message Queuing Telemetry Transport protocol”," *International Journal of Research,* Vols. ISSN: 2348-6848 Vol-3, Special Issue-3, p. 240, 27th January 2016.

[11] M. A. Tariq, "Non-functional Requirements in Publish/Subscribe Systems," *Ph.D. dissertation, Universit¨at Stuttgart, Fakult¨at Informatik, Elektrotechnik und Informationstechnik, Germany,* August 2013..

[12] M. Ion, G. Russello, and B. Crispo, "Supporting Publication and Subscription Confidentiality in Pub/Sub Networks," *Lecture Notes of the*

Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 50, p. 272–289, 2010.

[13] Muneer Bani Yassein, Mohammed Q. Shatnawi, Shadi Aljwarneh, Razan Al-Hatmi, "Internet of Things: Survey and open issues of MQTT Protocol," p. 1, 2017.

[14] Y. Chen and T. Kunz, "Performance Evaluation of loT Protocols under a Constrained Wireless Access Network,"," *International Conference on Selected Topics in Mobile Wireless Networking (MoWNeT),* 2016.

[15] Mark Austin, Michael Salmon,Joseph Dennisson,Michael Bates,Lawrence Ryan,Sheldon Meredith,, "DEVICE-DRIVEN INTELLIGENCE AND FEEDBACK FOR PERFORMANCE OPTIMIZATION AND PLANNING OF A SERVICENETWORK," p. 1, May 6, 2010.

[16] D. Locke, "MQ Telemetry Transport (MQTT) V3.1 Protocol Specification," pp. http://www.ibm.com/developerworks/library/ws-mqtt/, 2010..

[17] Luzuriaga, J. E., Perez, M., Boronat, P., Cano, J. C., Calafate, C., & Manzoni, P., "A comparative evaluation of AMQP and MQTT protocols over unstable and mobile networks. In Consumer Communications and Networking," *2015 12th Annual IEEE,* pp. 931-936, 2015.

[18] Muhammad Harith Amaran*, Nazmin Arif Mohd Noh, Mohd Saufy Rohmad, Habibah Hashim, "A Comparison of Lightweight Communication Protocols in Robotic Applications," *2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS 2015),* p. 400, 2015.

[19] Dipa Soni, Ashwin Makwana, "A SURVEY ON MQTT: A PROTOCOL OF INTERNET OF THINGS(IOT)," p. 1, 12 April 2017.

[20] Meena Singh, MA Rajan, VL Shivraj, and P Balamuralidhar, "Secure mqtt for internet of things (iot)," *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on. IEEE,* p. 746–751, 2015.

[21] X. Wang, J. Zhang, E. Schooler, and M. Ion,, "Performance evaluation of Attribute Based Encryption: Toward data privacy in the IoT," *Communications (ICC), 2014 IEEE International Conference,* p. 725–730, June 2014.

[22] Konglong Tang, Yong Wan ,Hao Liu,Yanxiu Sheng, Xi Wan and Zhiqiang Wei, "Design and Implementation of Push Notification System Based on the MQTT Protocol," *MQTT Protocol, instant messaging, Message Broker, Mobile Applications,* p. 116, 2013.

[23] Hwang, H. C., Park, J., & Shon, J. G., "Design and implementation of a reliable message transmission system based on MQTT protocol in IoT," *Wireless Personal Communications,,* pp. 1765-1777, 2016.

[24] Kannan Govindan and Amar Prakash Azad, "End-to-end service assurance in IoT MQTT-SN," *Consumer Communications and Networking Conference (CCNC),* p. 290–296, 2015.

[25] Whittaker S, Swanson J, Kucan J,, "Managing lightweight interactions in the desktop," *ACM Transactions on Computer-Human Interaction,* pp. 137-168, 1997.

[26] M. Ion, "Security of Publish/Subscribe Systems," *Ph.D. dissertation, University of Trento, Italy,* May 2013.

[27] Hsiang Wen Chen, Fuchun Joseph Lin, "Converging MQTT resources in ETSI standards based M2M platform," *2014 IEEE International Conference on Internet of Things (iThings 2014),* p. 292, 2014.

[28] Gero Mühl, "Large scale content based publish subscribe systems," *Ph D Thesis Darmstadt University of Technology,* 2002.

[29] Hwang, H. C., Park, J., & Shon, J. G., "Design and implementation of a reliable message transmission system based on MQTT protocol in IoT," *Wireless Personal Communications, 91(4),* pp. 1765-1777, 2016.

[30] Syaiful Andy, Budi Rahardjo2, Bagus Hanindhito, "Attack Scenarios and Security Analysis of MQTT Communication Protocol in IoT System," p. 1, 21 September 2017.

[31] "OASIS Message Queuing Telemetry Transport (MQTT) Technical Committee," *OASIS,* May 2014..

[32] Muhammad Harith Amaran*, Nazmin Arif Mohd Noh, Mohd Saufy Rohmad, Habibah Hashim, "A Comparison of Lightweight Communication Protocols in Robotic Applications," *2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS 2015),* p. 400, 2015.

[33] Jorge E. Luzuriaga, Marco Zennarot , Juan Carlos Cano, Carlos Calafate and Pietro Manzoni, "A Disruption Tolerant Architecture based on MQTT for loT Applications," *2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC),* p. 71, 2017.