Extension of Reception Frequency Control to MQTT Protocol

Kitae Hwang, Jae Moon Lee, In Hwan Jung

Abstract Background/Objectives: MQTT protocol is widely used for message delivery between sensors and small devices in IoT systems. When using MQTT, it is especially important to reduce unnecessary message delivery.

Methods/Statistical analysis: This paper proposes RFC(Reception Frequency Control) algorithm with which subscribers of the MQTT broker can control the maximum reception frequency of messages depending on their own processing ability and thereby the network traffic of the MQTT brokercanbe reduced.

Findings: The Mosquitto is an open source MQTT broker that implements the standard MQTT protocol. To verify the effectiveness of the RFC algorithm, we modified the Mosquitto to have the function of RFC and developed a test IoT system. We measured the network traffic of the existing Mosquitto and the modified Mosquitto with the RFC algorithm, respectively. The experimental results showed that the RFC algorithm reduces the message traffic of the MQTT broker extremely as well as keeps the network load of the MQTT broker uniformly.

Improvements/Applications: As a result, the RFC algorithm proposed in this paper not only enables low-capability devices or sensors to participate in MQTT based IoT systems,but also drastically reduces message traffic.

Keywords: IoT, MQTT, Message Broker, Publish-Subscribe, Mosquitto

I. INTRODUCTION

The MQTT is currently used as a message delivery broker of the IoTsystem[1-3]. It functions between the publishers and subscribers – users or devices that are distinguished accordingly to their roles. An enormous amount of messages is to be produced once smart cities, smart universities, or smart homes are formed, because IoT applications are composed of numerous sensors, devices, PCs, dashboards, and user devices[4-9].

An IoTsystem that utilizes the MQTT protocol is composed of an MQTT broker, subscribers, and publishers as can be seen in Figure 1[10]. What connects the subscriber with the publisher is a text data called as topic. A subscriber registers topics with the MQTT broker to receive messages regarding the topics. The MQTT broker saves such subscribers on a subscription list together with their topicsand manages them. When a publisher sends out a message of the topic to the MQTT broker, the MQTT broker then redirects the message to all subscriberswith the same topic as the message. The sending and receiving of **messageis** done through the MQTT broker, but it is the topic that determines which subscriberswill actually receive the message[11].

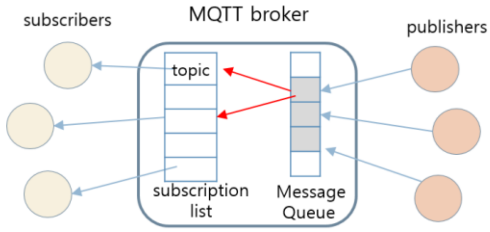


Figure 1.Messaging System with MQTT broker

On the other hand, if some sensors or devices, including user devices,publish data at a very high frequency, some subscribers can not catch up with the speed messagedelivery. Besides, there may be times when devices do not want to receive messages too often for some reasons. However, because the MQTTprotocol has no given standard for controlling the amount of messages received, the subscriber device is forced to receive all messages.

This paper aims to introduce theconcept of RFC(Reception Frequency Control) that enables the subscriber of the MQTT system to limit the receptionof messages under a given frequency level. This paper also extends RFC to the existing standard MQTT protocol while maintaining compatibility. Using RFC, the subscriber can specifythe maximum reception periodat which it wants to receive messages when it subscribesto the MQTT broker. This can reduce the network utilization of the MQTT broker as well as support devices with low hardware-capability.

In this paper, we modified the Mosquitto[12], an MQTT broker, that is provided by Eclipse as an open software to have the function of this RFC. We alsobuilt a test IoT systemandverifiedoperation and network performance of the modified Mosquitto with RFC function under the system.

II. MQTTWITH RFCALGORITHM

2.1. Concept of RFC

Every message is published with a topic by the publisher under MQTT protocol. Thesubscriber makes a packet consisting of a topic only to receivemessages from publishers andthen sends the packet to the MQTT broker. Let's add RFC function here. The subscriber chooses a specific time frequency at which it will receive messages and sends a packet including the frequency and the specific

Revised Manuscript Received on May 22, 2019.

Kitae Hwang, Jae Moon Lee, In Hwan Jung, School of Computer Engineering,Hansung University,Seoul, Korea
E-mail : calafk@hansung.ac.kr

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

message topic to the MQTT Broker.The time frequency means that the subscriber does not want to receivethe message more than the frequency and it does not care to receive messagesbelow the frequency. Therefore, the broker does not send more than one message per the time period, and the subscriber will receiveat most one message per the period of time. We call this frequency as the Maximum Reception Period(MRP) in other words.

Let me show a specific example of RFC on the current IoT system. In most cases of theIoT system, the main publishers are sensors. These sensors publish the sensor values on a constant cycle. Assume that there is a sensor that transmits the degree every second, and the topic is named 'degree'. Under the existing MQTT protocol, ifthe subscriber subscribes to the topic 'degree', it will receive a sensor value every second. When the RFC method proposed in this paper is applied, subscribers may designate a period of 5 seconds, so that the sensor value can be received at most onceevery 5seconds.

2.2.Extension of RFC function to MQTT Broker

To put RFC function into the existing MQTT protocol, three modifications must be done. First of all, the architecture MQTT of the broker should be modified. Second, when the subscriber subscribes to the MQTT broker, it should send a packet consisting of a topic and a value of maximum receptionperiod. Finally, the MQTT packet should be redesigned so that it can contain a value of maximum receptionperiod. In this section only how to insert RFC function the MQTT broker is explained and the others will be explained the later sections.

The architecture of anMQTT broker extended to include the RFC functionality is presented in Figure 2. The subscription list has been changed to have three additional fields,which are T_{period}, a value of maximum receptionperiod, T_{prev},the previous sending time, and a MSGfield that is a message buffer. In this paper, we use a time period(T_{period}) as a value of maximum receptionperiod. If T_{period} is 100 and the time unit is 10 milliseconds, it means that the subscriber does not want to receive more than 1 message per 1second. In other words, the MQTT broker should not send more than 1 message per 1second to the subscriber for a registered topic although more messages arrive in the Message Queue from publishers.

T_{prev} in the subscription list means the last time when the message has been sent to the subscriber. Of course the time unit of both T_{period} and T_{prev} should be the same, and the time unit may be 10 seconds, 1 second, or 1 millisecond, and so on.

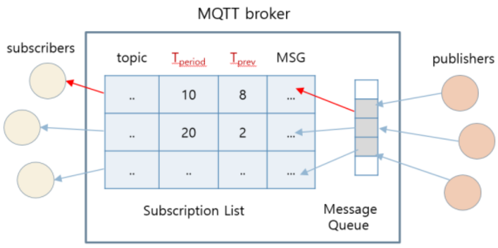


Figure 2.Internal of MQTT broker with RFC function

The last MSG field is a buffer to keep a message to be sent to the subscriber. The messages in the Message Queue is processed in two passes described in the Figure 3. The first pass is the process that the MQTT broker copies some messages from the Message Queue to MSG buffers of some entries of the subscription list. The MQTT broker checks allmessageswhich arrived into the Message Queue, according to Algorithm_RFC() described in the Figure 3.

Following Algorithm_RFC(), the broker traverses all messages through the Message Queue. Also for each message in the Message Queue, the broker compares the topic of the message with each topic of the subscription list one by one. For whatever subscription entry two topics are equal, the broker evaluates the two expressions of $T_{current}$ - T_{prev}>T_{period}, and T_{period} == 0, where$T_{current}$is the current time. We set up a way to accept the subscriberswho do not want to use the RFC function. T_{period} == 0 means that the subscriber does not use RFC function. The broker will have to send every message to the subscribers whose T_{period} == 0.So if the expression of line number 4 is true, the message is copied to the MSG buffer so that it can be delivered to the subscriber. Otherwise, since the time did not pass over the time period(T_{period}) designated by the subscriber, the message should not be sent to the subscriber. On the final step of the first pass, the broker removes all message from the Message Queue.

In the second pass, the broker sends the messages copied into MSG buffers at the first pass to the corresponding subscribers while checking through the subscription list. Whenever the broker sends the message in the MSG buffer, it changesT_{prev} field to $T_{current}$. And also it marks the MSG buffer as dirty.

```
Algorithm_RFC(input : Message Queue)

01: foreach message M in the Message Queue {
02:     foreach entry E in the subscription list {
03:         if topic of M == topic of E then
04:         if T_period == 0 or T_current - T_prev > T_period then
05:             copy the message M to the MSG buffer of entry E;
06:         endif
07:     endif
08:     }
09: }
10: remove all messages out of the Message Queue;
11: foreach entry E in the subscription list {
12:     if the MSG field is fresh then
13:         send the message in the MSG buffer to the subscriber;
14:         T_prev = T_current;
15:         mark the MSG buffer as dirty;
16:     endif
17: }
```

Figure 3. Algorithm for RFC function

2.3. MQTT PacketModification for RFC function

The standard MQTT packet is composed of a header and a payload. We decided to represent the value of Maximum ReceptionPeriod(MRP) by 2 bytes in the payload as shown in Table 1. When an MQTT client subscribes to the MQTT broker, it should put a value of MRP on the packet. The MQTT broker makes a new entry in the subscription list as shown in Fig. 2 and stores the value of MRP at the T_{period}field as well as a topic string. Asubscriber who does not want to RFC functionjust sends a packet that has a zero value at the RFC field.

Table 1: The MQTT Packet for RFC function

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Byte	0	1	2	3	4	5	6	7
Fixed Header	Message Type				Message Type			
	Remaining Length(1~4 bytes)							
Variabl e Header	Variable Header							
Payload	Topic							
	Subscriber QoS							
	Maximum ReceptionPeriod(MSB)							
	Maximum Reception Period(LSB)							

III. RESULTS AND DISCUSSION

3.1.Building a Test System

We have modified the Mosquitto with RFC function, where the Mosquitto is an MQTT brokerprovided by Eclipse as an open software.The Mosquitto runs as a single thread that processes all packets in only one loop. We have inserted some program codes somewhere in the loop with not changing the existing loop structure and not making additional threads. Also we have built a test IoT system to verify that RFC function. The test IoT system has a server PC that runs the modified Mosquitto broker with RFC function, a client PC that runs publisher applications, and another client PC that runs subscriber applications. In this paper we will call the existing Mosquitto broker as "MQTT standard" and the modified Mosquitto broker with RFC function as "MQTT-RFC", respectively in short.

3.2.Workloads

Workloads for experiments are shown in Table 2. We used only one topic and the fixed number of subscribers of 100for simplicity of experiments. But we conducted experiments with varying the number of publishers to 20 50, 100, 150 to 180. We also used 4 different valuesforthe MRPof the subscribers atthe ratesshown in Table 3.

Table 2: Workload Parameters

Parameter	Values
No. of Topics	1
No. of Subscribers	100
No. of Publishers	20, 50, 100, 150, 180
Message Size	100 bytes
Period Published by Publishers	10 messages/second

Table 3: MRPvalues for Subscribers

MRP values	Ratio(%)
5 seconds	30% of subscribers
2.5 seconds	30% of subscribers
1.25 seconds	30% of subscribers
0(Not usethe function of RFC)	10% of subscribers

3.3.*Experiment Results*

First, we conducted experiments to verify thatthe proposed RFC function works well. We simply modified the existing Windows application of "mosquitto_sub" to have a command argument of "-F MRPvalue" so that subscribers can designate a MRP value on subscription to the broker. And we set the unit of MRP as 10 milliseconds. So if you use 100 as a MRP value, it means 1 second(10x100=1000ms=1second).We ran the modified mosquitto_sub.exe with the command argument of "-F 100", which means 1 secondMRP, as you can see in Figure 4. The mosquitto_sub.exe connects to the MQTT-RFC broker and sends to it a packet composed of a topic of "degree" and a MRP value of 100.Figure 4 shows that asubscriber receives the messages regularly as their designated MRP of 1 second.

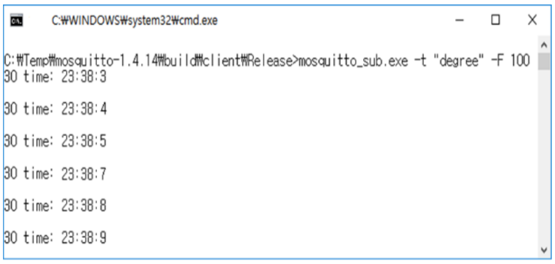


Figure 4. Console window of the Window PC which runs a mosquitto_sub.exe

Although the modified mosquitto_sub.exe connects to the MQTT standard broker not MQTT-RFC broker, no problem happens, because the MQTT standard broker will not interpret the payload part of the MQTT message. The "-F 100" command just does not work. This means that the application embedding the RFC method does not harm compatibility with applications implementing the standard MQTT protocol.Second, we compared the network traffic between the standard MQTT and the MQTT embedding RFC function.We measured network write bytesof the two brokers as performance index of network traffic. Figure 5 shows the result of the network traffic according to the number of publishers.

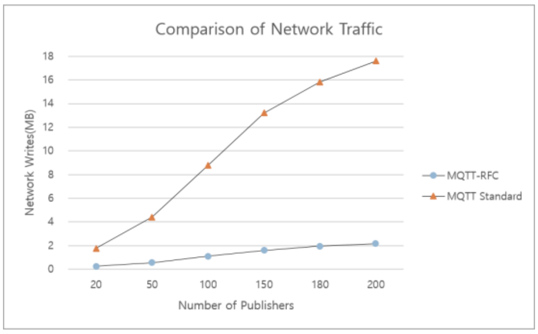
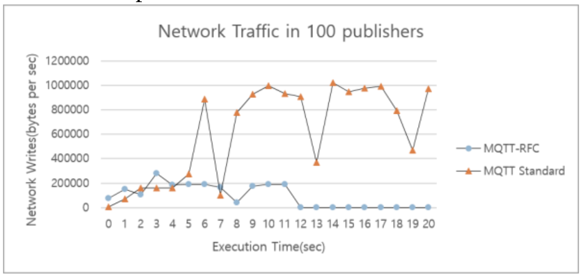


Figure 5. Comparison of network traffics

Figure 5 shows thatas the number of publishers increases, network traffic increases more, in the case of the standard MQTT broker. In contrast, in the case of the MQTT broker with RFC function, network traffic increases just very small.Figure 6 shows network write bytes and time spent of two MQTT brokers according to the number of publishers. The experiments were conducted that publishers send their messages to the broker only for 10 seconds. One can imagine easily that as the

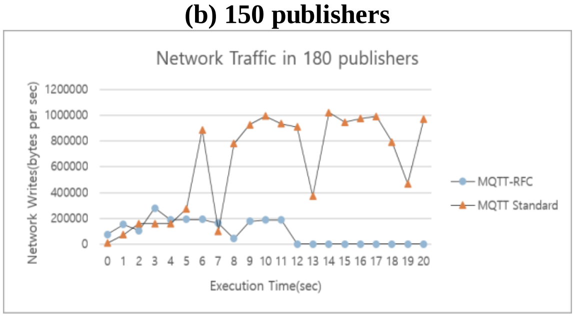
Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

number of publishers increases, not only the number of messages sent to the broker increases but also the number of messages sent to the subscribers from the broker increases. However, such a situation occurredonly in the standard MQTT broker as shown in Figure 6.The MQTT standard broker took more than 15 seconds to distribute to the subscribers the messages all that received from the publishers. On the other hand, the MQTT-RFC broker has completeddelivery to subscribers within 12 seconds regardless of number of publishers. This means that for the MQTT-RFC broker, it takes just 12 seconds to release the accumulated network load for 10 seconds but for the MQTT standard broker, it takes more time. As the number of publishers increases, it takes more time, as shown in Figure 6. In addition to the processing time, the MQTT standard broker costs about five times the network throughput. As a result, we can conclude from the experiment in Figure 6 that the proposed RFC methodcan reduce network traffic over the MQTT standard protocol.

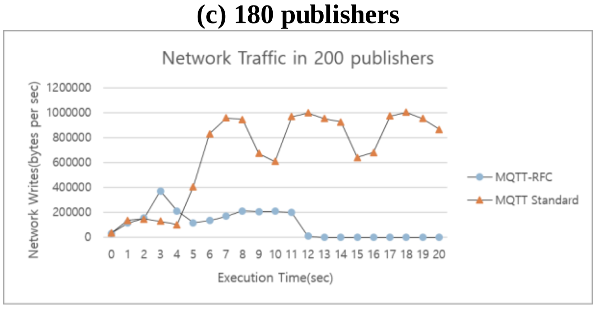


(a) 100 publishers

(b) 150 publishers



(c) 180 publishers



(d) 200 publishers

Figure 6. Comparison of network traffics according to the number of publishers

IV. CONCLUSION

Today's society is one where the IoT, a connection of small devices and sensors, is greatly used. In such a world, the MQTT protocol is widely used as a message broker between such sensors and small devices. In the perspective of the message broker and the devices, reducing the delivery of unnecessary messages is crucial for the performance enhancement of the overall system.This paper focuses on the MQTT protocol that is currently used to deliver messages between IoT devices and has proposed a concept of RFC(Reception Frequency Control),which is designed to control the frequency at which subscribers receive messages.The proposed method allowsthe IoT device to control to receive as many messages as can be stored and processed, so that theIoT device can participate in any IoT system regardless of its capacity. In order to verify the efficiency of the RFC function, the open source MQTT broker, Mosquitto, has been modified to have the proposed RFC functionality and a test IoT system has been built for experiments. Experimental results showed that,like our original goal, subscribers did not receive messages that exceeded the limitsspecified for the MQTT broker with RFC capability. They also showed that network traffic was much less when using the MQTT broker with RFC capability than the traditional MQTT broker.

ACKNOWLEDGMENT

This research was financially supported by Hansung University.

REFERENCES

1. Granjal J, Monteiro E,Silva JS. Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues.*IEEE Communications Surveys & Tutorials. 2015* thirdquarter:17(3):1294-312. doi: 10.1109/COMST.2015.2388550
2. Al-Fuqaha A, Guizani M, Mohammadi M, Aledhari M, Ayyash M. Internet of Things: A Survey on Enabling Technologies, Protocols and Applications. IEEE Communications Surveys &Tutorials. 2015 Fourthquarter;17(4):2347-76. DOI:10.1109/COMST.2015.2444095.
3. Yuan M. Explore MQTT and the Internet of Things service; 2017[Internet].[updated 2017Nov22; cited 2015Feb18].Available from:https://www.ibm.com/developerworks/cloud/library/cl-mqtt-bluemix-iot-node-red-app/index.html
4. Kodali RK,SoratkalS. MQTT based home automation system using ESP8266. IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Agra 2016 Dec;1-5. DOI: 10.1109/R10-HTC.2016.7906845
5. Sharma V, Tiwari R. A review paper on IOT & Its Smart Applications, International Journal of Science, Engineering and Technology Research. 2016 Feb:5(2): 472-76
6. Elhadi S, Marzak A, Sael N, Merzouk S. Comparative Study of IoT Protocols. Smart Application and Data Analysis for Smart Cities (SADASC'18); Available from: https://ssrn.com/abstract=3186315
7. Mishra B. TMCAS: An MQTT based Collision Avoidance System for Railway networks. Proc. of 18th International Conference on Computational Science and Applications (ICCSA). 2018 July 2. DOI: 10.1109/ICCSA.2018.8439562
8. Salman T. Networking Protocols and Standards for Internet of Things. [Internet]. [last modified 2015 Nov 30]. Availablefrom: https://www.cse.wustl.edu/~jain/cse570-15/ftp/iot_prot.pdf
9. Bandyopadhyay D, Sen J. Internet of Things: Applications and Challenges in Technology and Standardization. Wireless Personal Communications. 2011 May;58(1):49-69. DOI 10.1007/s11277-011-0288-5

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

10. Soni D, Makwana, A. A Survey on MQTT: A Protocol of Internet of Things(IOT). Proc. of International Conference on Telecommunication, Power Analysis and Computing Technique(ICTPACT). 2017 April. Available from :https://www.researchgate.net/publication/316018571_A_SURVEY_ON_MQTT_A_PROTOCOL_OF_INTERNET_OF_THINGSIOT(website)
11. Yassein MB, Shatnawi MQ, Aljwarneh, Al-Hatmi, R. Internet of Things: Survey and open issues of MQTT protocol. Proc. of 2017 International Conference on Engineering & MIS (ICEMIS). 2018 Feb 1. DOI: 10.1109/ICEMIS.2017.8273112
12. Eclipse Mosquitto. Mosquitto Open Source MQTT v3.1/v3.1.1 Broker. [Internet]. Available: http://mosquitto.org(website)