Traffic Prioritization in an MQTT Gateway

Tabinda
Department of Computer Science Engineering
Lovely Professional University
India

Nahita Pathania
Department of Computer Science Engineering
Lovely Professional University
India

ABSTRACT
It has not been much time since the Internet of Things (IoT) came into existence. It is a fresh concept that is always evolving. Ubiquitous computing, wireless technologies, sensing technologies, Internet Protocol (IP) and devices are mingled together in order to devise a system where the virtual or abstract world meets the real world meet and they interact continuously with each other. Wireless Sensor Networks on being integrated with IoT can work wonders. There are numerous sensors deployed in the sensor fields, all of them sending information generated by them towards an application in the cloud through an IoT gateway which helps to bridge the internal network of sensors with the World Wide Web. Different kinds of sensors forward different types of information towards the gateway. For instance there are temperature sensors that send information regarding temperature, sensors that send information about a patient's heart beat to the doctor and so forth. The objective of this paper is to prioritize the traffic/data generated by different sensors in a Message Queue Telemetry Transport for Sensor Networks (MQTT-SN) Gateway in order to mitigate the delay of data packets which is necessary for time critical applications.

Keywords
Internet of Things (IoT), Wireless Sensor Networks (WSN), Message Queue Telemetry Transport Protocol for Sensor Networks (MQTT-SN), Gateways, Traffic Prioritization.

1. INTRODUCTION
The Internet of Things (IoT) is a new and always evolving concept that consists of ubiquitous smart objects and devices with embedded sensors. The IoT embeds intelligence in the sensor devices to autonomously communicate, exchange information and take intelligent decisions[1]. The applications of IoT require access to data generated by sensors in real-time as per their needs. One of the very important components that are necessary for the realization of the IoT are the gateways. Gateways are meant to connect the sensor / device domain to the application domain. Gateways aggregate the real time sensor data dynamically and then this data is shared with the applications that need it over Internet. For example a mobile medical/ healthcare gateway gathers data from different sensors deployed for blood pressure, blood sugar, heart rate, etc. This data is then aggregated, filtered, analyzed and finally sent to the applications that need them [2]. The Internet of Things can be defined as a system that is comprised of huge networks of sensors, smart objects, etc that are connected to each other allowing communication with one another[3].

A WSN can be defined as a distributed system composed of a varying number of embedded devices, usually called nodes, provided with a processing unit, a wireless communication interface, and a set of sensors/actuators, making these devices capable of sensing real physical environment or interacting with it[4]. A typical wireless sensor network is made up of a huge number of sensors as well as actuators which are battery-operated and have limited amount of storage and processing capabilities. There are hundreds and thousands of sensors of different types deployed in the sensor field. These sensors generate information of different types. For instance there are dedicated sensors for recording temperature, humidity, pressure, heart rate of a patient, etc.

Since a huge number of sensors and actuators are deployed, the devices must be connected wirelessly otherwise a huge cost will be incurred on connecting them with the help of wires. WSN's are quite dynamic in nature as the wireless links may break at any point of time and therefore lead to the replacement of nodes. Because of their dynamic nature, wireless sensor networks do not use the conventional methods of using addresses for the purpose of communication. Moreover, doing so would cause an overhead as there are a large number of sensor- actuator (SA) devices involved in a wireless sensor networks.

A significant problem that WSN's have to face is regarding the addressing schemes of networks involved. This can however be overcome by means of a data centric communication protocol in which the receivers get information not based on their network addresses but on the basis of data content. One well-known example of data-centric communication protocol is Publish/Subscribe. Using the Publish/Subscribe method along with WSN's makes the field data collected by the SA devices available to all the applications.The Internet of Things and the wireless sensor networks provide different data centric protocols so as to access data of embedded devices and the sensor networks. These data centric protocols are of two types: request-response and publish-subscribe. The Publish and Subscribe protocol is very popular nowadays because here the same sensor data can be routed and published to more than one final processing units which inturn are connected to the router gateways.

Message Queue Telemetry Transport , which is a light-weight, open source publish/subscribe protocol can be used for this purpose. The problem with MQTT is that it uses TCP/IP which is very complex for low-power and simple devices like wireless SA's. In order to overcome this problem

Message Queue Telemetry Transport-Sensor Networks (MQTT-SN), which is a version of MQTT is used. MQTT-SN is an MQTT variant that is meant for sensor networks.

In this paper we put forward a proposal to prioritize data in a Message Queue Telemetry Transport (for Sensor Networks) Gateway in order to mitigate the delay of data packets which is necessary for time critical applications. Since there are hundreds and thousands of sensors deployed in sensor fields,

all these sensors will pass the information collected by them to the MQTT-SN gateway in order to send this information to the respective applications residing in the cloud. We will use aggregating MQTT-SN gateways in which only one connection is established with MQTT broker.

2. GENERAL DISCUSSION

2.1 Message Queue Telemetry Transport Protocol (MQTT)

Its main job is to collect data generated by devices. As its name suggests the main purpose of this protocol is telemetry or in other words remote monitoring. It is a publish/subscribe messaging transport i.e. it simply lets the receivers also called subscribers let the publisher know that they are interested and the receiver or publisher in turn stores their addresses in order to know where to send which message.

Message Queue Telemetry Transport protocol is extremely light weight and is used to connect small devices to constrained networks. It collects device data and transfers the same to IT infrastructure. It is used by applications like Facebook Messenger. Since one does not want to lose data, so this protocol runs over TCP, which ensures reliability.

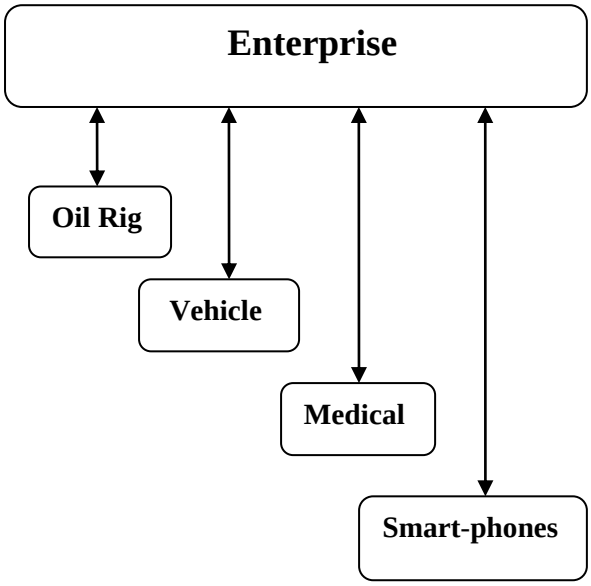

Figure 1: Message Queue Telemetry Transport Protocol

MQTT ensures reliability by providing:
i) Fire and forget: Here message is only sent once and it does not require any acknowledgement [5].
ii) Delivered at least once: In this case message is sent at least one time and also requires an acknowledgement. When the QoS level is equal to one, the protocol makes sure that a message arrives at the server at least once [6].
iii) Delivered exactly once: Four way handshake mechanism is used to ensure that the message is delivered exactly once.
MQTT finds its use in various applications, for instance to monitor a huge oil pipeline in order to check leaks or any kind of vandalism, power usage monitoring, intelligent gardening, lighting control, etc. MQTT has a hub-and-spoke architecture [7].

MQTT (formerly the MQ Telemetry Transport) is light weight protocol that was basically designed and devised for connecting devices with power constraints over low-bandwidth networks. Andy Stanford-Clark and Arlen Nipper are the two men who originally designed this protocol. They were assigned a responsibility to invent a different protocol in order to connect oil pipelines over networks that were not reliable. MQTT protocol does not have adequate security features. It just employs a user-password authentication in the name of security. Moreover, it does not have any kind of an authorization mechanism [8].

2.1.1. Concept

MQTT uses the publish/subscribe pattern to connect parties that are interested in communicating with each other. The publisher (sender) sends a message to a particular topic for which a number of subscribers (receivers) are waiting in order to receive the message. The subscribers as well as the publishers are autonomous; they therefore do not need to be aware of each other's presence.

2.1.2 Components of MQTT

Client – A client can be any publisher or subscriber which connects itself to a broker rather, a centralized broker over a network. It is important to note that MQTT has both servers and clients. Clients can either be persistent or they can be transient. A client's session with the broker is maintained by the persistent clients while the broker does not track the transient clients.

Broker – The software that receives all the messages from the clients that act as publishers and forwards them towards the clients that act as subscribers. Since the broker can result in a single point of failure or become the bottleneck, it is therefore clustered for the purpose of scalability and reliability. Broker is the entity that ensures that the data from publishers reaches the receiving clients known as subscribers [9].

Topic – Topics are endpoints to which the different clients connect. Topics are quite simple strings that are hierarchical in nature and are encoded using UTF-8, delimited by a forward slash. Topics are case sensitive. Topics have two levels:

i) Single level: building /+/ humidity

* building /kitchen/ humidity

* building /living room/ humidity

ii) Multiple level (only at the end): building / room/ #

* building /kitchen/wall/ temperature.

* building /livingroom/ceiling/ humidity.

2.2 Message Queue Telemetry Transport for Sensor Networks (MQTT-SN)

Message Queue Telemetry Transport for Sensor Networks (MQTT-SN) is an extension of the open source Message Queue Telemetry Transport (MQTT) protocol. It follows the design developed to use on the top of TCP/IP protocol. Originally it was called MQTT-S, with the S concept of MQTT, focusing in allowing operations on low-cost and low-power sensor-actuators devices, most of the integrated in non-TCP/IP networks. The main usage of this protocol is to provide a simple and scalable communication meanwhile allowing a seamless integration of the WSN into the traditional networks. In a MQTT-SN system, the running applications and devices can be both Publisher and Subscribers. The published message always passes in the Broker, even if they both reside in the same network. The construction of the widespread topics is based in hierarchical scheme. It supports the usage of three levels of Quality of

Service (QoS). The communication between the devices/clients inside the WSN with the traditional network is made through the Broker. This protocol allows more than one running gateway, providing more robustness.

MQTT SN has two types of components: MQTT-S clients and MQTT-S gateways (GWs). MQTT-S clients are on the Wireless Sensor Network side and they enable the Sensor-Actuator devices to access the publish/subscribe services of an MQTT broker. They connect to the gateway using the MQTT-SN protocol, and the gateway in turn connects to the broker. The main function of the gateway is to act as an interpreter between the MQTT and MQTT-S protocols. It is not necessary for an MQTT-S gateway to be always integrated with the broker. The case in which the gateway is not integrated into the broker, it uses the MQTT protocol in order to communicate with the broker.

2.2.1 General Message Format of MQTT-SN

A MQTT-SN message is made up of two parts:
i) A header whose length is 2 or 4 octets.
ii) A variable part that is optional.

The header part of the message is always present and contains the same fields but the message's variable part is dependent upon the type/kind of the considered message.

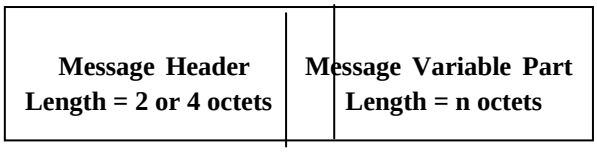

Fig 2: General Message Format

2.2.1.1 Message Header

Its format is given below:

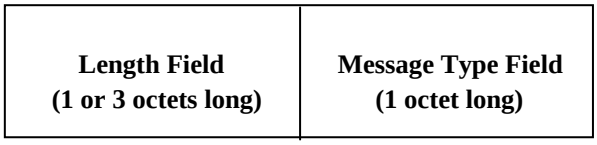

Fig 3: Message Header Format

i) Length:

The length of this field is either 1 or 3 octets and it gives the total number of octets in the message. Fragmentation and reassembly of messages is not supported by MQTT-SN, therefore the maximum packet size supported by the network controls the maximum length of messages that can be used in a network.

ii) Message Type:

The length of Message Type field is 1-octet and specifies the message type. Some of the message types are:

i) CONNECT: The client sends the CONNECT message in order to setup a connection.

ii) CONNACK: Server sends this message in response to a client's request for connection.

iii) PUBLISH: Clients as well as gateways use this message in order to publish data for a particular topic.

iv) PUBACK: A gateway or a client sends this message to acknowledge the receipt and processing of a PUBLISH message provided the QoS level is either 1 or 2.

v) SUBSCRIBE: Clients use the SUBSCRIBE message so as to subscribe to a particular topic name.

vi) SUBACK: A gateway sends the SUBACK message to a client in order to acknowledge the receipt and processing of a SUBSCRIBE message.

2.2.1.2 Message Variable Part

The type of the message determines the content of the message variable part. Message variable part has the following fields:

i) ClientId: Just like MQTT, this field contains a 1-23 character long string that is required to uniquely identify a client to the server. This field is of variable length [10].

ii) Data: This field is analogous to an MQTT PUBLISH message payload. The application data that is to be published is contained in this field. This field also has a variable length.

iii) Duration: The length of this field is 2-octets. It gives a time period's duration in seconds.

iv) Flags: The Flags field is 1-octet and contains the following flags:

- DUP: This flag is set to "0" if a message is sent for the first time. If however the message is retransmitted then it is set to 1.
- QoS: Three QoS levels:

i) Level 0- At most once. Zero is the minimal level and it guarantees a best effort delivery. In this case a message won't be acknowledged by the receiver or stored and retransmitted by the sender. This is often called "fire and forget" and provides the same guarantee as the underlying TCP protocol.

ii) Level 1- At least once. When using this level of Quality of Service, it is ensured that a message will be at least delivered one time to the receiver. But it can be also delivered more than once.

iii) Level 2- Exactly once. The highest QoS is 2, here it is guaranteed that each message is received only once by the counterpart. It is the safest but also the slowest level of quality of service.

- Retain: same meaning as with MQTT i.e. this message indicates whether the flag saves the latest message for a particular specified topic which as the last known good value. When new clients will subscribe to that topic they will receive the last retained message on that topic immediately after subscribing.
- Will: If this flag is set then it indicates that a Will topic is being asked for by a client.
- CleanSession: same meaning as with MQTT
- TopicIdType: This field specifies whether the field TopicId present in this message has a normal topic id , short topic name, etc.

v) GwAdd: The GwAdd (Gateway Address) field contains the address of a GW and has a variable length

vi) GwId: The length of GwId (Gateway ID) field is 1-octet and it is used to identify a gateway uniquely.

vii) MsgId: Here the sender is permitted to match a message with the acknowledgment corresponding to it.

viii) ProtocolId: This field is present in a CONNECT message only and is analogous to the MQTT 'protocol version' and 'protocol name'. It is a 1-octet long field.

ix) Radius: The value of the broadcast radius is indicated by this field and it is 1-octet long.

x) TopicId: The value of the topic id is contained in this field and its length is 2-octets.

xi) TopicName: The TopicName field gives the topic name and has a variable length.

xii) WillMsg: This field contains the Will message and has a variable length.

xiii) WillTopic: This field contains the Will topic name and has a variable length.

3. LITERATURE REVIEW

The authors did a systematic review of papers but could not find any work related to this topic. They first searched the internet for papers by using the keyword MQTT-SN Gateway and it fetched about 116 papers. The authors then searched by using keyword MQTT-SN Gateway in Internet of Things which fetched 98 papers. Then to further narrow down their search, the authors used the keyword Aggregating MQTT-SN Gateway in Internet of Things and got about 37 papers. Even out of these 37 papers none was related to the topic being discussed i.e. no work was done on prioritizing traffic in an MQTT Gateway. Some of those 37 papers along with the papers on Internet of Things that have been reviewed are as follows:

3.1 Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges

This paper has discussed the evolution of internet towards the present day Internet of Things and how just human-human communication have been transformed into human-human, human-device and device-device communication. This paper also presents five layer architecture of Internet of Things. Some applications have been highlighted like prediction of natural disasters, smart cities, smart homes, industry applications, etc.

Certain key challenges that are being faced by Internet of Things have been discussed. Some of them are:

1. Naming and Identity Management- Since Internet of Things connects innumerable objects, each object should have unique identity over the internet. Therefore an efficient mechanism for naming and identification of things is required.

2. Standardization and Interoperability-There are many vendors that provide technologies and services that are not accessible by others. So standardization is a must for interoperability of all devices in Internet of Things.

3. Information Privacy-Internet of Things uses different enabling technologies like RFID, 2D and as all types of daily use objects will carry tags for identification that will embed the information about a particular object, privacy must be insured.

3.2 A Survey on Application Layer Protocols for the Internet of Things

This paper discusses the protocols that are used for communication purposes in Internet of Things at the application layer. Different protocols have been studied and compared in order to know how well they are suited for Internet of Things on the basis of factors such as how much energy they consume, reliability, etc.

Several factors influencing the selection of protocols at the application layer have been identified and the most important of them are: consumption of battery, computational speed, ability to communicate with other devices.

3.3 Architecture of Things and its key Technology Integration based on RFID

This paper puts forward the concept of six layer architecture of Internet of Things which has been derived from the working flow of Internet of Things and the basic three layer architecture. The six layers as proposed in this paper are-coding layer, information acquisition layer, information access layer, network layer, information integration layer and application service layer. This paper also presents automatic recognition system which is based on Zigbee by integrating wireless sensor networks (WSN) and radio frequency identification (RFID).

3.4 MQTT-S – A Publish/Subscribe Protocol for Wireless Sensor Networks

This paper throws light on MQTT-S, a modification of the Message Queue Telemetry Transport protocol in order to meet constraints of Wireless Sensor Networks.

The publish/subscribe standard meets many of the requirements for the purpose of WSNs communication since it can hide the network topology and permit delivery of data on the basis of an individual device's interests rather than its address. An important advantage of MQTT-S over other Internet of Things protocols is that MQTT-S is based on a well-established publish/subscribe protocol already widely used. Implementation of MQTT-S has brought forth many challenges of WSNs. The implementation also shows that this protocol can be easily implemented on devices that have only limited resources.

3.5 Securing Smart Maintenance Services: Hardware-Security and TLS for MQTT

According to this paper a secure end to end connection is required between deployed devices and the remote maintenance service provider for the process of remote data acquisition for smart maintenance services. In this paper the authors studied a use case of AVL Particle Counter (APC) and the MQTT Information Broker (MIB) and then investigated the client authentication problem in order to secure the Message Queue Telemetry Transport protocol. The proposed design utilizes TLS concept in order to append a secured layer of communication underneath Message Queue Telemetry Transport protocol. The proposed system also utilizes a hardware security controller which performs client authentication by means of TLS.

3.6 An´alise de Desempenho deBrokers MQTT em Sistema de Baixo Custo (Performance Analysis of MQTT Brokers in Low Cost System)

This paper presents a performance analysis (CPU usage, memory consumption and message throughput) of MQTT brokers in a low-cost hardware, the Raspberry Pi 2 Model B. The objectives of the analysis are to ascertain which MQTT broker implementation is best suited to the limitations of the hardware and to verify if the Raspberry Pi 2 is actually able to function as a gateway in a sensor and actuator network for the internet of things (IoT). The results showed that the Raspberry Pi 2 can handle large number of connections and that the implementation in Erlang (eMQTT) obtained the results in data throughput, while the implementation in C obtained the lowest CPU load and memory consumption.

3.7 Multi-Protocol Transport Layer QoS: An Emulation Based Performance Analysis for the Internet of Things

This paper demonstrates that wisely chosen transport protocols can increase the efficiency of resource usage of a network under specific network conditions. Selecting real time transport protocols in real time makes possible the achievement of a distributed embedded system having different actors capable of reacting to application specified Quality of service as well as varying network conditions. vNET, which is a custom, visualization based, distributed network emulation test bed has been presented as well as validated using an Message Queue Telemetry Transport (MQTT) performance analysis before it was used to validate the premise of multi-protocol transport layer QoS.

3.8 Lightweight Internet Protocols for Web Enablement of Sensors using Constrained Gateway Devices

Lightweight Internet protocols are nowadays greatly being used in ubiquitous environment in order to optimize the usage of resources of constrained devices such as a smart mobile gateway. This paper puts forward a study on the different such protocols in order to optimize network resources, the usage of energy, and computation cost of a constrained gateway device. Feature wise categorization and comprehensive analysis of existing dominant protocols, such as MQTT (message queue telemetry transport), CoAP (constrained application protocol) have been provided so as to achieve improved understanding of the existing issues as well as gaps in this domain. This paper also identifies the best suited application areas for each protocol on the basis of results corresponding to the typical requirement of resources as well as performance attributes.

3.9 Toward better horizontal integration among IoT services

This paper throws light on the major shortcomings of the current IoT protocols and also suggests a rule-based intelligent gateway with the help of which the gap between existing IoT protocols will be bridged in order to enable the effective integration of horizontal IoT services. This

intelligent gateway does enhance the protocol fragmentation in IoT context but does not address the cause of fragmentation. This paper proposes an enhanced MQTT protocol version that mitigates the problems prevalent in the existing MQTT protocol. No work has been done on MQTT gateway to prioritize traffic.

3.10 Internet of Things: A Survey on Enabling Technologies, Protocols and Applications

This paper gives an overview of the Internet of Things (IoT) including its protocols, enabling technologies and application issues. A thorough summary of the application issues and protocols has been provided along with some of the key challenges that are being faced by Internet of Things. The authors have also discussed big data analytics, fog and cloud computing in context of Internet of Things.

3.11 A Scalable and Sustainable Web of Buildings Architecture

The authors discuss how Web technologies in context of smart buildings are beneficial to make the application level homogenous, resulting in intelligent/smart and reusable entities. Emphasis has also been laid on how the REST architectural style can be applied to al the levels, in order to homogenize the entire ecosystem. It also discusses the MQTT gateways, both transparent as well as aggregating. But no work has been done on prioritizing traffic on these gateways.

3.12 IoT integration on Industrial Environments

This paper proposes that the performance of IoT devices that are inserted into an IP backbone including industrial environments may be proved by IoT smart gateways without degrading the network load. An important feature of this IoT gateway is the usage of selected data structures along with the implementation of deadband models. The data that is sent across shared IP networks can therefore be organized, selected and concentrated with the help of the IoT gateways. The results of simulation performed using Networked Control Systems show that it is feasible to apply these models to the gateway.

4. PROBLEM DEFINITION AND OBJECTIVES

Internet of Things is a new and always evolving concept in which different devices are interconnected and can interact with each other without the need of human intervention .The applications of IoT require access to data generated by sensors in real-time as per their needs. One of the most important components that are required for realization of the IoT are the gateways. Gateways are meant to connect the sensor / device domain to the application domain. A typical wireless sensor network consists of hundreds and thousands of sensors that generate huge traffic and forward it to their respective applications in the cloud via a gateway. As the number of sensor nodes grows the blocking delay (or queuing delay) suffered by packets of some MQTT-SN nodes may be high. However, for time-critical applications this delay is not tolerable. As such, a prioritization of MQTT-SN nodes is required such that the nodes which high priority are blocked for least amount of time. We propose to build a prioritized packet scheduler in an MQTT gateway, which shall mitigate this delay.

5. PROPOSED METHOD

Step 1: Collect information:

In this step, the focus is on collecting the information about the MQTT gateway (GW), MQTT protocol specification, and creation of basic without priorities) aggregating MQTT gateway model in Matlab.

Step 2: Create packet-delay graph:

The main factor that needs focus when simulating and analyzing basic MQTT GW mode is delay. From the randomly generated traffic an average delay graph of MQTT-SN nodes packets is to be found. This graph will be used later to draw a comparison between basic MQTT-GW and prioritized MQTT-GW.

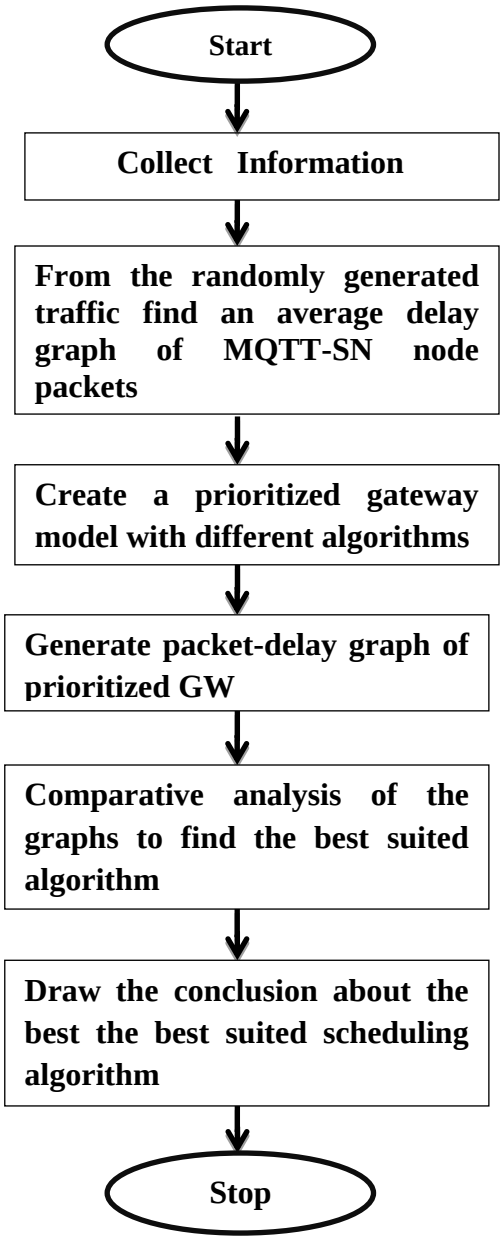

Fig 4: Data Flow Diagram of Proposed Method

Step 3: Create Prioritized GW model:

In this step, three types of GW each with a unique packet scheduling algorithm, like FIFO and Round Robin (RR) need to be created.

Step 4: Generate packet-delay graph of prioritized GW:

Here delay graphs for each GW of step 3 are created, in the same manner as was done in step 2.

Step 5: Comparative analysis:

Here the graphs generated in step 4 are compared with the graphs generated in step2 to see how each scheduling algorithm in a MQTT-GW impacts the delay of priority nodes.

Step 6: Summary:

Draw conclusion about the best suited scheduling algorithm.

6. EXPECTED OUTCOMES

The expected solution will try to prioritize data in an MQTT gateway. This will be done by addition of priorities to MQTT packet streams by using different priority scheduling algorithms such as First in First out (FIFO) and Round Robin (RR) on MQTT packets, in an MQTT gateway. Delay is the main factor needs to be taken into account, when simulating and analyzing basic MQTT gateway model.

7. CONCLUSIONS AND FUTURE SCOPE

In today's world, where data is continuously being generated and transmitted over the networks, it is necessary to categorize data on the basis of its importance compared to other data. The proposed system aims to prioritize data in an MQTT-SN gateway so that time critical applications e.g. data from sensors recording seismic changes get more priority can be given higher priority as compared to data from a temperature sensor. As a future work, a study of gateway can be conducted, as a gateway can become a bottleneck for message traffic, or become a single-point-of-failure. This problem may be mitigated by adding redundant gateway devices, which can result in scalability and reliability (to avoid bottlenecks and single-point-of-failures).

8. REFERENCES

[1]. Khan, R., Khan S U., Zaheer R., and Khan, S. Future internet: the internet of things architecture, possible applications and key challenges. Frontiers of Information Technology (FIT), 2012 10th International Conference on. IEEE.

[2]. Bandyopadhyay, S., and Bhattacharyya, A. Lightweight Internet protocols for web enablement of sensors using constrained gateway devices. Computing, Networking and Communications (ICNC), 2013 International Conference on. IEEE, 2013.

[3]. Niemi, J. The design and implementation of sensor communication protocol with connectivity adapter interfaces in nRF51822 embedded development platform. 2016.

[4]. Mottola, L., and Picco, G. P. Programming wireless sensor networks: Fundamental concepts and state of the art. Computing Surveys (CSUR) 43.3 (2011): 19.

[5]. Karagiannis, V., Chatzimisios, P., Gallego, F.V., and Zarate, J.A. A Survey on Application Layer Protocols for the Internet of Things. Transaction on IoT and Cloud Computing 3.1 (2015): 11-17.

[6]. Luzuriaga, J.E., Cano, J.C., Calafate, C., Manzoni, P., Perez, M., and Boronat, M. Handling mobility in IoT

applications using the MQTT protocol. Internet Technologies and Applications (ITA), 2015. IEEE, 2015.

[7]. Schneider, S. Understanding the protocols behind the internet of things. Electronic Design, http://electronicdesign. com/iot/understanding-protocols-behind-internet-things, October 09, 2013.

[8]. Niruntasukrat, A., Issariyapat, C., Pongpaibool, P., Meesublak, K., Aiumsupucgul, P., and Panya, A. Authorization mechanism for MQTT-based Internet of Things. Communications Workshops (ICC), 2016 IEEE International Conference on. IEEE, 2016.

[9]. Hunkeler, U., Truong, H. L., and Clark, A.S. A publish/subscribe protocol for Wireless Sensor Networks. Communication systems software and middleware and workshops, 2008. Comsware 2008. 3rd international conference on. IEEE, 2008.

[10]. Clark, A.S., and Truong, H .L. MQTT For Sensor Networks (MQTT-SN) Protocol Specification . International business machines (IBM) Corporation version 1 (2013).

[11]. Zhang, M., and Xu Cheng, F.S. Architecture of Internet of Things and its Key Technology Integration Based-on RFID . Computational Intelligence and Design (ISCID), 2012 Fifth International Symposium on. Vol. 1. IEEE, 2012.

[12]. Lesjak, C., Hein, D., Hofmann, M., Maritsch, M., Aldrian, A., Priller, P., Ebner, T., Ruprechter, T., and Pregartner, G. Securing smart maintenance services: Hardware-security and TLS for MQTT. *Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on. IEEE, 2015.*

[13]. Torres, A. B.B., Rocha, A.R., and De Souza, J.N. Análise de Desempenho de Brokers MQTT em Sistema de Baixo Custo.

[14]. Wilcox, J., Kaleshi, D., Sooriyabandara, M. Multi-Protocol Transport Layer QoS: An Emulation Based Performance Analysis for the Internet of Things. International Journal on Advances in Intelligent Systems, vol 7 no 3 & 4, year 2014.

[15]. Fuqaha, Ala Al, Abdallah Khreishah, Mohsen Guizani, Ammar Rayes, and Mehdi Mohammadi. "Toward better horizontal integration among IoT services." *IEEE Communications Magazine*, September 2015. doi:10.1109/MCOM.2015.7263375.

[16]. Bovet, Gérôme. "A Scalable And Sustainable Web Of Buildings Architecture". N.p., 2017.

[17]. Fuqaha, Ala Al, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. "IoT integration on Industrial Environments." *IEEE Communications Surveys & Tutorials*. doi:10.1109/COMST.2015.2444095.

[18]. Fuqaha, Ala Al, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. "Internet of Things: A Survey on Enabling Technologies, Protocols and Applications." *IEEE Communications Surveys & Tutorials*. doi:10.1109/COMST.2015.2444095.