Performance of Message Queue Telemetry and Transport (MQTT) Protocol for Internet of Things (IoT) with Symmetric Encryption and Varying Cipher Blocks

Atul Oak
Ph.D. Research Scholar
Dept. of Electrical Engineering
Veermata Jijabai Technological Institute (VJTI)
Mumbai, India

R.D. Daruwala
Professor
Dept. of Electrical Engineering
Veermata Jijabai Technological Institute (VJTI)
Mumbai, India

ABSTRACT
Internet of Things (IoT) is a network of interconnected physical devices used to share the data between the devices. IoT system connects the physical world of devices like sensors with the digital world of computing. Billions of devices are already connected to the internet and huge amount of data is being shared between these devices. IoT system uses special protocols called machine to machine protocols (M2M) for sharing of data between the devices. Message Queue Telemetry and Transport (MQTT) is one of the most widely used, light weight application layer M2M protocol used in IoT systems. Devices and networks used in the design IoT systems have lots of constraints like use of low end eight bit microcontrollers working limited memory like Random Access Memory (RAM) with limited computing power and use of unreliable network. To work with such constrained devices and networks, light weight protocols like MQTT have been designed. It is said that the data is new oil and every data that is shared on the network is vulnerable to different types of threats. Security of data is a very basic requirement for any IoT system. MQTT protocol being a light weight protocol does not directly support much security mechanism for the data shared between the devices. This research is focused on improving the security aspects of MQTT protocol and to provide an efficient and general security solution around MQTT protocol. This paper presents some part of the experimental results of the research based on measuring the performance of MQTT protocol with standard symmetric block encryption.

General Terms
Internet, Protocol, Network, Data, Experiment.

Keywords
Internet of Things, MQTT, Architecture, Symmetric encryption, Block cipher, Performance.

1. INTRODUCTION
Next step after internet is called Internet of Thing (IoT) in which physical devices like sensors also called as things can talk to each other electronically [1]. For last two decades, internet has been widely used for the electronic communication between two persons by using computing machines and such communication is called person to person (P2P) communication. When internet is used for sharing data between two physical devices, it called thing to thing communication (T2T). It is estimated that already more than 20 billion devices are connected to the internet and are sharing tons of data [2]. When such billions of connected devices are sharing the data through internet, one of the very basic requirement and concern is a security of data that is shared between the two devices. Special protocols called light weight protocols are used in the IoT systems for sharing of data between the two devices. The primary reason of using light weight protocol is that the IoT systems normally operates within a very constrained environments like low end eight bit microcontroller with limited computational capability, limited power, limited memory, limited available bandwidth (BW) and use of unreliable networks for sharing of data. Message Queue Telemetry and Transport (MQTT) is one of the most common IoT protocol designed in year 1999 as a M2M protocol specially to work with devices working in such constrained environments. MQTT protocol was designed as a lightweight protocol to operate in the constraint environment and in an attempt to make it as light weight protocol; it lacks many of the security features required for secure sharing of the data between the devices.

2. LAYERS IN IoT SYSTEM
IoT systems are normally implemented as layered systems. Though there is no standard architecture for IoT systems, one of the common layered architecture is called Service Oriented Architecture (SOA). It defines four different layers [4] [12] as shown in the table 1.

Table 1: Different Layers of IoT System

Layer No.	Layer Name	Function of the Layer
1	Perception Layer	Collection the data
2	Network Layer	Routing of the data
3	Service Layer	Services to the data
4	Application layer	Delivery of the data

The four layers are as follows:

1. Perception layer: it is a physical layer of IoT system which is used to collect the data using different types of devices like sensors.
2. Network Layer: Routing of the data collected by a physical device is done using network layer of IoT system.
3. Service Layer: It is considered as an interface layer which provides different types of services to the application layer.
4. Application Layer: User interacts through application layer and it is final destination for data delivery.

3. PROBLEM STATEMENT
IoT systems are usually designed with lots of constrained devices and works in constrained environment. IoT system may

be designed with basic eight bit or sixteen bit micro-controller with limited resources and computing power and may be deployed in the unreliable networks with low bandwidth. Many light weight protocols are specially designed to work with such constrained devices and constrained environment. MQTT is one of the most widely used light weight protocol at the application layer of the IoT system. Being light weight protocol, MQTT protocol do not support many security features and do not provide reliable security while sharing the data. Such compromised data can be easily visible and accessible by using different sniffing tools like wire-shark. Such data is also vulnerable to the different types of threats like sniffing attack. Most of the security for MQTT based IoT application is provided using Transport Layer Security (TLS) protocol based on Secure Socket Layer (SSL) protocol. TLS protocol is not been designed for IoT systems and it is a heavy weight protocol for IoT systems which works with lots of constraints. Also, TLS protocol does not provide end to end security for MQTT based applications. It is identified that end to end payload encryption is a better option for sharing of data using MQTT protocol [11]. It is a need of time to improve the security features of MQTT protocol based applications. This research aims to evaluate the performance of MQTT protocol with different standard encryption techniques and further design a light weight and efficient security solution for MQTT protocol to enhance the security features of MQTT protocol. Once the performance of MQTT protocol with standard encryption techniques is evaluated, this research will propose security architecture for MQTT protocol.

4. APPLICATION LAYER PROTOCOLS

Application layer is topmost layer in IoT system. It works like an interface between the end devices with the underlying network and the user. In IoT systems, this layer is used for data formatting and presentation. Literature study indicates that five protocols [3] [7] [6] are most commonly used for sharing of data at the application layer of the IoT system. MQTT protocol was designed by IBM as an asynchronous publish and subscribe type of M2M protocol especially for use with constrained devices [5]. In MQTT protocol, client devices do not send data directly to each other but send data to a central server called MQTT broker. A routing technique called a topic [14] is used in MQTT protocol to identify client which receives the data. The MQTT protocol runs on the top of the Transmission Control Protocol (TCP) and Internet Protocol (IP) called TCP/IP stack. MQTT protocol supports a feature called quality of service (QoS) which can be used depending on the application. There are three types of QoS levels supported by MQTT protocol called QoS0, QoS1 and QoS2. Common IoT applications of MQTT protocol are home automation systems like light control, power monitoring and energy monitoring, constrained networks, medical applications and smart homes [10]. Hyper Text Transfer Protocol (HTTP) is a type of synchronous request and response protocol commonly used for internet applications like World Wide Web (www) also called browsing. This protocol may not be suitable for the IoT system working with lots of constraint because HTTP protocol adds lots of overhead or redundancy. It proves to be a heavy weight protocol for IoT systems with lots of constraints. The Constrained Application Protocol (CoAP) is also synchronous request and response protocol [5]. This protocol was designed by the Internet Engineering Task Force (IETF). The CoAP unlike MQTT protocol runs on the top of the User Datagram Protocol (UDP) and it may not be suitable for unreliable networks very commonly used in the IoT systems. In comparison, UDP based CoAP communication may have lower overhead than TCP based MQTT communication but CoAP based communication may get degraded due to possible packet losses since it do not support technique like TCP retransmission. It is further observed in literature study that MQTT and CoAP are most widely used application layer protocols in IoT systems and MQTT protocol may become a de facto standard for IoT systems [9]. The Advanced Message Queuing Protocol (AMQP) is commonly used protocol in applications based on financial industries and banking. It is observed that JPMorgan, an American banking and financial services company used AMQP to send almost 1 billion messages per day [8]. AMQP is an asynchronous publish and subscribe messaging protocol. It also runs on the top of the TCP like MQTT protocol but now a day this protocol is not used widely. The Extensible Messaging and Presence Protocol (XMPP) was also standardized by the IETF. This protocol was designed for near real-time communication applications like chatting and message exchange [5]. It also runs on the top of TCP and it supports publish-subscribe and request-response type of messaging system. It does not support a facility like QoS. XMPP is based on extended markup language (XML) messages and it also creates more overhead or redundancy. Hence XMPP may not be suitable for IoT system working with lots of constraints.

5. COMPARISON OF COMMON APPLICATION LAYER PROTOCOLS

Basic comparison between the five most common application layer protocols in IoT system [3] [12] is shown in the table 2. The comparison of different protocols indicates that MQTT is one of the light weight protocol with minimum header size and hence widely used in IoT applications

Table 2: Comparison of five Application layer protocols in IoT

Protocol	RESFUL Architecture	Transport	Publish/ Subscribe	Request/ Response	Security	QoS	Header size in Bytes	Standard
CoAP	Yes	UDP	Yes	Yes	DTLS	Yes	4	IETF
MQTT	No	TCP	Yes	No	SSL	Yes	2	OASIS
XMPP	No	TCP	Yes	Yes	SSL	No	XML	IETF
AMQP	No	TCP	Yes	No	SSL	Yes	8	OASIS
HTTP	Yes	TCP	No	Yes	SSL	No	JSON/XML	IETF

6. SECURITY AND MQTT PROTOCOL

MQTT protocol was specially designed as a lightweight protocol with minimum focus on the security features for data [11]. There are three pillars of security called Confidentiality, Integrity and Availability and it is called a CIA model of security. Data privacy or confidentiality in MQTT protocol is a major concern since MQTT protocol does not directly provide any support for data encryption [13]. Security of data in MQTT protocol is mainly dependent on lower layer transport protocol

called Transmission Control Protocol (TCP) using a mechanism called SSL or TLS. SSL or TLS protocols are commonly for applications like browsing on internet using HTTP protocol on personal computers (PC's). But, this mechanism is not suitable for constrained devices used in the design of IoT system like use of low end microcontroller. Also, TLS mechanism does not provide end to end security in MQTT protocol. Different types of data encryption techniques can be used with MQTT protocol to provide an end to end encryption of data that is electronically communicated between MQTT clients [11]. End to end encryption may avoid the dependency on TLS and the associated problems due to various constraints. Data encryption techniques can be classified into two basic types called symmetric encryption and asymmetric encryption. Symmetric encryption technique uses only private key between the two MQTT clients for encryption and decryption of data. Asymmetric encryption technique uses private key along with public key between the two MQTT clients for encryption and decryption of data. Asymmetric encryption technique is computationally more resource incentive than symmetric encryption and hence may not be suitable for constrained devices used in the IoT system. Symmetric encryption techniques can be further classified into two types called block ciphers and stream ciphers. In block ciphers, plain text or data is divided into the larger blocks of fixed size and each block is encrypted separately into a cipher text. In Stream cipher, encryption is performed on the plain text bit by bit to generate the cipher text. After literature study, research was focused on three types of block cipher algorithms called Data Encryption Standard (DES), Advanced Encryption Standard (AES) and Blowfish. AES is most commonly used block cipher which is extension of DES while Blowfish is more suitable for constrained devices. There is several code blocks available with block ciphers for encryption. During research, experimentation is performed to measure the performance of MQTT protocol using standard symmetric block encryption techniques like DES, AES and Blowfish and by varying the code blocks for the encryption. In Electronic Code Book (ECB) mode, data is divided into the fixed size blocks and each block is encrypted independently. ECB is like a raw cipher in which for each block of the input we get the corresponding encrypted output. ECB may not be the most preferred mode for the encryption of data since same data pattern in the block may have similar encryption patterns which can be recognized. This mode is less secure since pattern becomes predictable and data becomes vulnerable to the attacks. In Cipher Block Chain (CBC) mode, each block of plaintext is EX-ORed (XOR operation) with the previous block of plaintext and the result of XOR operation is finally encrypted into a cipher text. CBC has an initialization vector EX-ORed with first block of plain text. It is observed that CBC is one of the common modes used for encryption but the process of encryption looks serial and it may require more time. Propagating Cipher Block Chaining (PCBC) mode is an extension of CBC mode in which each block of plain text is EX-ORed with previous block of plaintext and the cipher text and result of XOR operation is finally encrypted into cipher text. It is observed that this technique is not so commonly used in the applications. Cipher Feedback mode (CFB) is a derivative of CBC mode which may work on block cipher as a stream cipher. Output Feedback Mode (OFB) uses key stream block to XOR with the plain text and perform encryption and to get cipher text.

7. EXPERIMENTAL TOOLS AND METHODOLOGY

For measuring the performance of MQTT protocol with the standard symmetric block encryption techniques, the following software tools were used for the experimentation

1. Mosquitto (version 3.1.1) as a MQTT broker running on a computing machine. It is a server for MQTT based communication.
2. Eclipse Neon (.3) Integrated Development Environment (IDE) for MQTT publisher client.
3. MQTTfx (v1.5.0) for MQTT receiver client.

The system specifications of the computing machine used for the experimentation and running different software tools is as follows:

1. CPU running at 1.5GHz clock frequency.
2. Memory (Random Access Memory) of size 2GB.
3. Windows 7 as an Operating System.

Three symmetric block encryption techniques called DES, AES and Blowfish were implemented with MQTT protocol in a java language. TCP port 1883 is used for connecting MQTT client as a publisher with MQTT broker and MQTT client as a subscriber with MQTT Broker. TCP PORT 1883 is standard unencrypted port for MQTT protocol used for communication of data without TLS mechanism. Data is received by MQTT client subscriber using end to end encryption with DES, AES and Blowfish encryption techniques.

8. RESULTS OF EXPERIMENT

Results of our experiments are shown in the Table 3. The time complexity measurement shows the time required to execute different standard symmetric encryption techniques with MQTT protocol for transmitting the data. For each encryption technique, ten samples were used to find the time to encrypt the data using MQTT protocol. The average time for encryption of data is calculated using above samples for specific encryption technique and the results [14] are shown in the table 3. Each encryption technique is measured for different Quality of Service (QoS) level used with MQTT protocol like QoS0, QoS1 or QoS2 for sharing the data. Experimentation is done under all identical conditions.

Table 3: Performance of MQTT protocol with block encryption

Encryption technique for 1 KB data	Average time for encryption in milli-seconds (mS)
DES (QoS = 0)	819
DES (QoS = 1)	821
DES (QoS = 2)	822
AES (QoS = 0)	850
AES (QoS = 1)	851
AES (QoS = 2)	852
Blowfish (QoS = 0)	552
Blowfish (QoS = 1)	553
Blowfish (QoS = 2)	554

Figure 1 shows the performance of MQTT protocol with the standard symmetric block encryption techniques. This measurement of the time is done under normal conditions with operating system's (OS) systems processes running normally along with publisher, subscriber and broker.

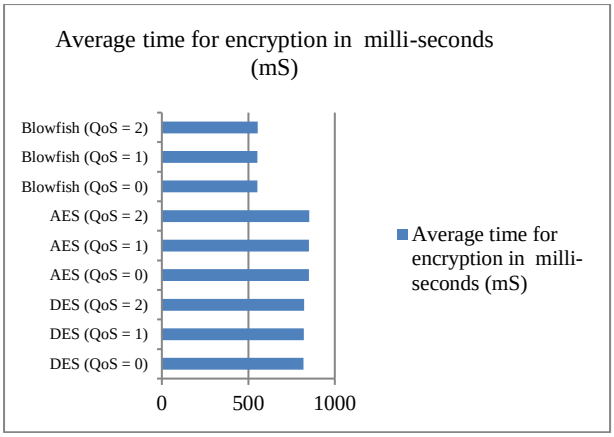

Fig. 1: Response of MQTT protocol with standard encryption techniques

From the measurements, it is observed that Blowfish encryption technique provides comparatively faster response compared to other encryption techniques. Experiment was performed further with Blowfish encryption technique by changing the size of Blowfish encryption key for sharing the data. Table 4 shows the effect of varying the size of a key in Blowfish encryption.

Table 4: Effect of change in size of a key in Blowfish encryption

Sr. No.	Size of the key in bits	Time to execute in Milliseconds (mS)
1	32	552
2	64	612
3	96	650
4	128	711
5	160	750
6	192	775
7	224	798
8	448	824

Figure 2 shows the effect of changing the size of a key in Blowfish encryption technique for sharing the data using MQTT protocol. This time is also measured with OS systems processes running normally along with publisher, subscriber and broker.

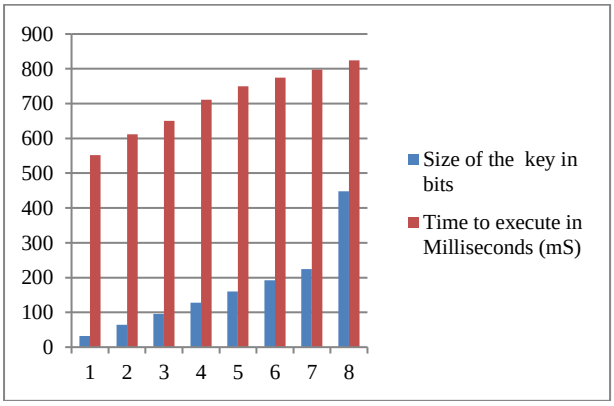

Fig. 2 Response of MQTT protocol with change in key size

of Blowfish encryption

Experiment was further extended with Blowfish encryption technique and by using various types of cipher blocks for encrypting the data. Table 5 shows the response of different types of cipher blocks for encryption of data with MQTT protocol and Blowfish encryption.

Table 5: Response of Blowfish encryption with different cipher blocks

Sr. No.	Cipher Block – 1 KB Data	Average in milliseconds
1	CBC (QoS = 0)	650
2	ECB (QoS = 0)	675
3	CFB (QoS = 0)	680
4	OFB (QoS = 0)	677
5	PCBC (QoS = 0)	670
6	CBC (QoS = 1)	651
7	ECB (QoS = 1)	677
8	CFB (QoS = 1)	681
9	OFB (QoS = 1)	678
10	PCBC (QoS = 1)	671
11	CBC (QoS = 2)	652
12	ECB (QoS = 2)	678
13	CFB (QoS = 2)	682
14	OFB (QoS = 2)	680
15	PCBC (QoS = 2)	672

Figure 3 shows the effect of different types of cipher blocks for encryption of data with Blowfish encryption. This time is also measured with OS systems processes running normally along with publisher, subscriber and broker.

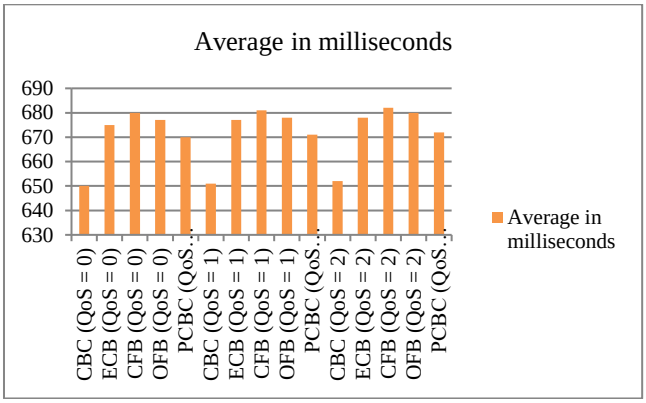

Fig. 3 Response of Blowfish encryption with different cipher blocks

9. CONCLUSION

MQTT is one of the most widely used light weight application layer protocol in the IoT systems for sharing the data between the two devices. MQTT protocol does not provide reliable security features. MQTT protocol depends on lower layer protocols like TLS for the security of data. TLS proves to be a heavy weight security solution for MQTT and it does not provide end to end security. Light weight and robust security solution is required for MQTT protocol. Through experimental results, the performance of the MQTT protocol is measured with standard symmetric block encryption techniques like DES, AES and Blowfish. MQTT protocol is programmed individually with standard symmetric block encryption techniques like DES, AES and Blowfish for sharing data from the MQTT publisher to the MQTT subscriber. Time complexity required to encrypt the data is measured. Results show that Blowfish encryption technique provides comparatively faster response with lowest time to encrypt the data. For every encryption technique, it is also observed that QoS do not have a very significant effect on the performance of encryption. There is a little change in the performance of MQTT protocol with change in the size of a key using Blowfish encryption technique. It is observed that there is a variation in the performance of MQTT protocol with Blowfish encryption with variation of different cipher blocks. All these performance results will be further used for proposing light weight and robust security architecture for MQTT protocol.

10. ACKNOWLEDGMENTS

Our thanks to Veermata Jijabai Technological Institute (V J T I) Mumbai and Vidyalankar Institute of Technology (VIT), Mumbai for providing required facilities for our research.

11. REFERENCES

[1] Jatinder Singh, Thomas Pasquier, Jean Bacon, Hajoon Ko, and David Eyers, Twenty Security Considerations for Cloud- Supported Internet of Things, IEEE Internet of Things Journal, Vol. 3, No. 3, June 2016.

[2] Kwok-Yan Lam and Chi-Hung Chi, Identity in the Internet- of-Things (IoT): New Challenges and Opportunities, Springer International Publishing AG 2016

[3] Mehdi Mohammadi et al. “Internet of Things: A Survey on Enabling Technologies, Protocols and Applications”, article in IEEE Communication Surveys and AMP Tutorial, January 2015.

[4] Jie Li, Wei Yu, Nan Zhang, Xinyu Yang, Hanlin Zhang and Wei Zhao, A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications, IEEE Internet of Things Journal 2016.

[5] Vasiteious Karagianmis et. al., A survey on Application Layer Protocols for IoT, transaction on IoT and Cloud Computing 2015, ISSN: 2331-4753 print.

[6] Stefan Mijovic, Erion Shehu and Chiara Buratti, Comparing Application Layer Protocols for the Internet of Things via Experimentation, 2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow.

[7] Jorge Granjal, Edmundo Monteiro and Jorge Sa’silva, Security of Internet of Things: A survey of existing protocols and open research issues, IEEE communications surveys and tutorials 2015.

[8] Aimaschana Niruntasukrat et.al. Authorization Mechanism for MQTT based IoT, IEEE 2016 workshop on IoT.

[9] A Rizzardi, et al., “AUPS: An Open Source Authenticated Publish/Subscribe System for IoT”, Information System (2016), http://dx.doi.org/10.1016/j.is.2016.05.004

[10] Anusha M et. al., Performance Analysis of Data Protocols of Internet of Things: A Qualitative Review, International Journal of Pure and Applied Mathematics, volume 115, No. 6 2017, ISSN: 1311-8080(print version)

[11] Edielson et. al., M2M protocols for constrained Environments in the context of IoT: A comparison of approaches, Dec.2015

[12] Atul Oak, R.D. Daruwala, Comparison of major Applications Layer Protocols in Internet of Things, Research Colloquium, International Conference on Communication, Information and Computing Technology – ICCICT 2018

[13] Syaiful Andy et. al., Attack Scenarios and Security Analysis of MQTT Communication Protocol in IoT System, Sept 2017

[14] Atul Oak, R.D. Daruwala, Assessment of Message Queue Telemetry and Transport (MQTT) Protocol with Symmetric Encryption, International Conference on Secure Cyber Computing and Communication” IEEE 2018.