International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-6, March 2020

Side-by-Side Analysis of Key Exchange Algorithms for Load Optimized Security in MQTT Protocol

Suja P Mathews, Raju R Gondkar

Abstract*: *The Information Technology has evolved and we have reached at an era of Internet of Things (IoT). According to International Telecommunication Union's Global Standards Initiative (GSI), it is the network of all kinds of 'things' embedded with electronics, sensors, actuators, software etc. Things connected in open Internet poses high security risks. Majority of these devices use Message Queuing Telemetry Transport (MQTT) protocol for exchanging information. Most of the devices with limited storage and computing power are connected using MQTT. Since the protocol doesn't provide any mechanism for encryption, the security aspect of the protocol is really weak. This paper describes the need for empowering security in MQTT. In this research work, we benchmark different cryptographic algorithms and propose the best possible algorithm to enable higher level of security in MQTT. This work further demonstrates how to use the proposed algorithm to enable lightweight key exchange mechanism among MQTT devices.

Keywords: Cryptographic Algorithms, Lightweight Key Exchange, MQTT, Public key, Payload encryption, Security.

I. INTRODUCTION

As per the predictions and forecast, the Internet in future will be capable of connecting around 25 billion things. Every 'thing' will hold unique identification and can be easily accessed by the network. IoT is associated with a number of protocols. Among all of these, MQTT is the most suitable protocol for low power devices. Millions of devices are connected to Internet using MQTT and they communicate and exchange information [1]. Enabling security for each and every information exchange is a big challenge. The applications of IoT like connected cars, smart homes, cloud-connected devices, healthcare, wearable, smart city etc. demand a higher level of security. Many of such devices rely on MQTT protocol. The limited authentication capability and absence of message encryption are the main drawbacks of this protocol. Microcontrollers like ESP8266-12E, which cannot hold up the heavy requirements of Transport Layer Security (TLS) because of its less storage capacity. The Machine-to-Machine (M2M) communication protocols built on these types of small microcontrollers are vulnerable to attacks like eavesdropping, denial of service and unauthorized access. Lack of security can cause leak out of confidential data or eavesdrop on your information by third parties. This paper studies the possible threats on MQTT protocol and the work focuses on identifying a proper algorithm to improve the security aspects of MQTT and thereby provide powerful and trusted devices to IoT environment.

The rest of the paper is organized as follows. Section II describes the review of literature done in order to identify the possible algorithms for message encryption in MQTT. Section III elaborates the scope for research work and defined the problem with research objectives. Section IV is briefing about the security vulnerabilities associated with MQTT protocol. Section V discusses in detail about several symmetric and asymmetric algorithms. Section VI finalizes an apt algorithm to implement payload encryption in MQTT. Finally, Section VII concludes the work and explains the scope for future work.

II. LITERATURE REVIEW

While extremely useful for low power, low cost and reliable messaging systems [2], the MQTT protocol is low on security aspects. This is practically the only messaging protocol, that can be used on low power and low cost devices like ESP-8266 Wi-Fi chip, which is a very popular microcontroller in the IoT world, especially for on-premise device communication. Being used within organization, there is a chance of confidential information being transmitted to the servers from devices, and such transmissions need to be adequately secured. Further, the current method of implementing security for existing deployments will need full replacement of the devices with more powerful devices to support encryption. There is no mechanism to enable security for the existing devices with simple software upgrades.

The study has done on the best journal publications to achieve the aims of the present research work by surveying the past research. The literature review has focused on the various cryptographic algorithms, which can help to enable payload encryption in publish/subscribe protocol [3] like MQTT. The review of literature demonstrates that there are symmetric as well as asymmetric algorithmic approaches for encrypting and decrypting the data. The literature review identified with a number of widely used algorithms like DES, AES, RSA, DSA and ECC. This research paper does a comparative study on these algorithms to find the highly efficient and best possible algorithm for cryptography.

Manuscript received on February 10, 2020.
Revised Manuscript received on February 20, 2020.
Manuscript published on March 30, 2020.
* Correspondence Author
Suja P Mathews*, Department of IT, Jain University, Bangalore, India. Email: sujam.mail@gmail.com
Dr. Raju R Gondkar, Department of PG Studies, CMR University, Bangalore, India. Email: rrgondkar@gmail.com

III. SCOPE OF RESEARCH WORK

The initial study has tracked down the pros and cons of MQTT and a strong need to enhance the security aspects of the protocol. A survey has conducted on the different cryptographic algorithms to get contemporary ideas on encryption techniques. The essential information and concepts gathered were used as a starting point for a deeper investigation of research work and to make possible improvements.

A. Problem Description

In an IoT environment various types of devices are inter-connected. Exchanging information takes place between multiple devices. This information has to be secured by payload encryption. Presently MQTT protocol does not have a proper mechanism for authorization and encryption. Providing payload encryption is the main challenge while using MQTT. The fixed and variable headers of MQTT control packets [4] are not providing any mechanism to encrypt the payload. An analysis has to be done to find out the most suitable algorithms for lightweight key exchange in MQTT protocol. The protocol must ensure that message is encrypted and secured throughout the communication.

B. Research Objectives

This paper discusses about the security vulnerabilities and privacy concerns associated with MQTT protocol. Based on the literature survey, an initial study has conducted on the security of MQTT protocol [5]. The study helped to identify and set the objectives for the research work. The primary objective of this paper is to identify the various algorithms available for payload encryption. In order to choose an apt methodology for lightweight key exchange mechanism, we need to do a comparative analysis on the identified algorithms.

IV. SECURITY VULNERABILITIES

MQTT is an ideal protocol for developing M2M (Machine to Machine) communication. In an IoT environment, the front-end devices receive data through smart sensors. All of these sensors or devices may not be under monitoring system and the possibility for vulnerabilities to attack is too high. Unauthorized access to data, eavesdropping and denial of service are the few possible threats that can cause major issues and affect the security and privacy. The protocol has a very simple security model. It currently allows a username and optional password for authentication. Data has to be encrypted and protected from third party's eavesdrop. Since multiple devices connected to the system, a huge volume of data will be sending across the network. This heavy traffic can also cause denial of service attacks.

For encryption and transport-level security, the Transport Layer Security (TLS) standard is recommended, although this heavy TLS stack is not always suitable for small devices. In order to secure the information, we need to rely on cryptographic techniques. MQTT handles with small handheld devices with limited memory. Light cryptographic algorithms will be suitable for payload encryption.

V. STATE OF ART ANALYSIS

MQTT protocol is a lightweight communication protocol. It can be run on low power, low bandwidth and less storage capacity devices. In this protocol, the publisher, broker and the subscriber exchange information and communicate with each other. To provide a mechanism to communicate securely is the main concern with MQTT. There are number of cryptographic techniques which are commonly used. The usage of cryptographic algorithms should not slow down the protocol performance. This paper gives an overview of some popular symmetric and asymmetric public key protocols used for key exchange mechanisms.

A. Symmetric Algorithms

Symmetric algorithms make use of a common private key for encryption and decryption of message. The secret key is exchanged between sender and receiver in advance. Since the key is kept secret between the sender and the receiver, symmetric key cryptography is also called as secret key cryptography. Here the sender encrypts the message using the private key. The encrypted message is known as cipher text. At the receiver side, the cipher text is decrypted with the common private key as shown in Fig. 1. Symmetric algorithms are fast and efficient [6]. Creating unique key for each pair of devices is the biggest challenge. When cryptography involves n number of devices, n (n-1)/2 keys have to be generated to encrypt and decrypt the messages.

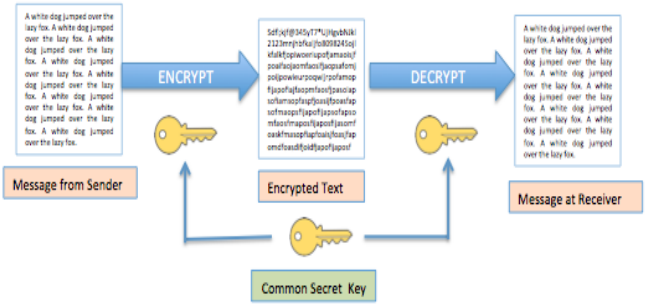

Fig. 1. Symmetric Key Cryptography

- **Data Encryption Standard (DES)**

The Data Encryption Standard is asymmetric key 64-bit block cipher algorithm. Algorithm needs 56-bit keys for encryption. Using these 56-bits, it is possible to generate 256 keys. DES provides sixteen rounds of encryption and each round makes use of a different key. It is based on the substitution-permutation network. The DES function uses P-boxes for transposing the bits and S-boxes substitutes the bits and thereby generating a cipher. Properties like completeness and avalanche effect make this algorithm very strong. The concept of "Triple DES" uses a key bundle, which consists of three DES keys, K1, K2 and K3, each of 56 bits. Usage of 56-bits key is the biggest disadvantage of DES. Weakness identified is that, sometimes if you choose two inputs to a S-box, it creates the same output. In 1999, it took less than a day to decrypt the DES encrypted message. It was proven inadequate and less secure against the cyber attacks and not allowed to use after 2023.

Advanced Encryption Standard (AES)

Advanced encryption Standard is one of the most commonly used block cipher algorithm. National Institute of Standards and Technology (NIST) developed this algorithm in 2000.Itis a fast and secure encryption based on substitution, shift and bit-mixing techniques.

This is 128-bit block cipher algorithm. Algorithm can use 128,192 or 256-bit encryption keys. This is mathematically more efficient. AES encryption is faster and provides much more security when compared to DES [7]. This algorithm requires more processing and needs more rounds of communication as compared to DES. This in turns slowdowns the performance of the algorithm. There are certain limitations and drawbacks for AES. Encryption of every block of data is in the same way. Implementation of this algorithm with software is literally hard and complex.

B. Asymmetric Algorithms

Asymmetric key algorithms are also called as public key cryptography. Here both sender and receiver use two separate keys, one public key and one private key for encryption and decryption. Though public and private keys are different, these are linked mathematically. The public key will be shared among the devices. Sender converts the message into cipher text by using the public key of the receiver. On the other end, the encrypted text is decrypted using the private key of the receiver as shown in Fig. 2. Asymmetric algorithms are more secure and robust and less vulnerable to security breaches. At the same time, it requires complex computation for generating two different keys as public and private keys.

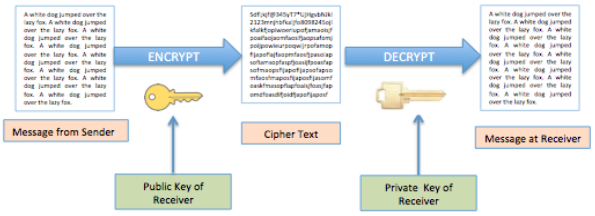

Fig. 2. Asymmetric Key Cryptography

- **Rivest-Shamir-Adleman (RSA)**

RSA (Rivest-Shamir-Adleman) is an asymmetric public key cryptography algorithm. It operates on two different keys as shown in Fig. 3. One public key, which is known to all clients and the other one, is kept private. The strength of encryption depends on the size of the key. This algorithm uses keys of 1024, 1092 or 2048-bits long. Here the key is calculated by taking the modulus of the product of two prime numbers. Finding out the original prime numbers from this result or factoring is too complex and infeasible. This makes the algorithm safe and secure. RSA uses much large integer keys for generating the cipher. The number of possible keys that can be created is too high. So hacking the cipher by brute force attacks would not be possible. For communications with higher threats, algorithm uses keys of 2048 or even 4096-bits. In this case, key generation becomes extremely slow. Usage of larger size keys is the main drawback of RSA.

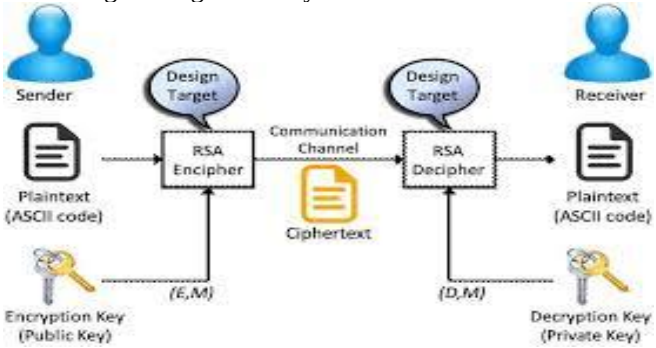

Fig. 3. Structure of RSA Algorithm

- **Digital Signature Algorithm (DSA)**

National Institute of Standards and Technology (NIST) developed digital Signature Algorithm (DSA) in 1991. DSA was considered as an alternate to RSA algorithm. Algorithm uses a pair of numbers of 160-bits. Using a unique and complex mathematical function, DSA generates digital signature [8]. This digital signature is attached along with the message at the start of data transmission. Only the authorized sender using their private key can create the signature. The signature can be verified using a public key, which is known to everyone. In fact DSA is only used to generate signatures for authenticating messages and it doesn't provide any option for encryption of data. The computational time for the signature is less and the length of the generated signature is small in size. So DSA requires less storage for computations. There are disadvantages for this algorithm. The message in DSA is only authenticated by the creator and not encrypted. Also it takes lot of time to authenticate the messages.

- **Elliptic Curve Cryptography (ECC)**

The Elliptic Curve Cryptography is intrinsically an asymmetric encryption method. Neal Koblitz and Victor S. Miller introduced this algorithm in 1985. The functions and the properties of elliptic curve theory are the basis for ECC. Usually in traditional methods, the product of large prime numbers generates the key. But ECC makes use of the properties of elliptic curve equation to create keys.
This algorithm allows both sender and receiver to set up a private key. Here the two parties exchange public keys computed by them using different private keys [9]. This algorithm creates smaller cryptographic keys, which is faster and efficient. The computational overhead of ECC is very less. Like every other algorithm, ECC provides an equivalent higher level of security to each block of data. It requires less battery usage as well as low computing power. Since ECC provides a better performance with a smaller key, it got widely accepted. Emerging technologies use this algorithm especially for mobile applications.

C. Comparison of ECC with RSA and DSA

The main purpose of message encryption is that unauthorized users do not get knowledge of original message. Most implementations leveraging public key cryptography [10] for encryption use RSA, and are also used for Digital Signature. Secure RSA needs a longer key, and has been increasing over the years. These longer keys, in turn put a larger overhead on programs using RSA. Such larger overheads have potential to impact sites with large number of secure transactions, like ecommerce sites. Hence, many implementations have started leveraging elliptic curve cryptography (ECC) instead of RSA.ECC offers equivalent security levels with a much smaller key size, and this in turn, reduces the overheads significantly. Table-1 gives the ratio in key-size, security bits and ratio of cost measured for ECC in comparison with RSA/DSA [11].

Table 1: Comparison of ECC with RSA/DSA

Key Size		Security Level	Ratio of Key Size	Ratio of Cost
RSA/DSA	ECC			
1024	160	80	7:1	3:1
2048	224	112	10:1	6:1
3072	256	128	12:1	10:1
7680	384	192	20:1	32:1
15360	521	256	30:1	64:1

The performance is evaluated based on the above table.It describes the key size used by algorithmsRSA,DSA and ECC for encryption. The RSA/DSA algorithm offers security level for 128-bits data by generating a key of size 3072-bits . At the same time, ECC provides same security level with a much smaller key with size 256-bits [12]. There is a huge difference in the ratio of key size and cost. This proves the importance and advantage of ECC over RSA/DSA. Fig. 4 illustrates the difference in key size for RSA/DSA and ECC for various security levels.

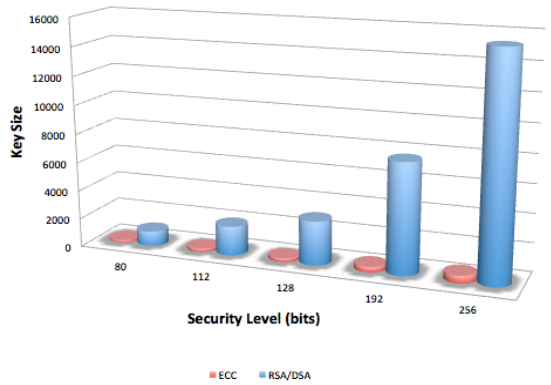

Fig. 4. Key size comparison for various security levels

VI. ECC FOR KEY EXCHANGE IN MQTT

Comparison has carried out on both symmetric and asymmetric algorithms. The study describes the importance of asymmetric algorithms over symmetric. The main drawback of symmetric approach is that it has to exchange single key between devices. ECC is really fast in key generation and able to provide encryption and decryption with the help of smaller keys. This reseach work proved that ECC is much more efficient and faster than the other algorithms.

Fig. 5 illustrates the key exchange mechanism using Elliptic Curve Diffie-Hellman algorithm for MQTT protocol. This explains how key exchange takes place between the device (publisher) and the broker. Public key P_A and P_B is calculated using the private key of publisher and broker at both sides. These public keys are exchanged between the sending device as well as broker. The secret key (N_A . P_B) and (N_B . P_A) are generated using these exchanged public keys, where N is a secret number picked by the sender and receiver. Later the shared key K=N_B. (N_A.G), is calculated with the help of these secret keys where G is the base point. Since the secret key computed by both parties result in an identical key, we can extend this approach to leverage it to encrypt and decrypt the payload.

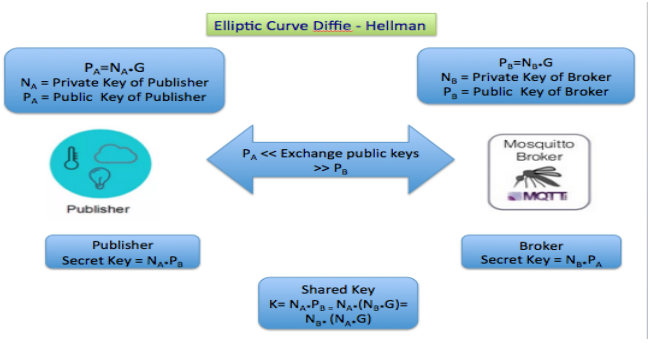

Fig. 5. Key exchange in MQTT using ECC

VII. CONCLUSIONAND SCOPE FOR THE FUTURE WORK

The research work has focused on various symmetric and asymmetric algorithms and has analyzed the advantages as well as disadvantages of each algorithm. Both symmetric and asymmetric algorithms perform encryption in different ways. Symmetric algorithm uses a single key and that has to be shared among all clients who need that message. Asymmetric algorithm requires two keys, one public and other private key for encryption and decryption of data. Symmetric encryption exchanges the key among clients and this may weaken the security. On the other hand, asymmetric encryptions need both public and private key. It is considered to be slow but significantly more secure. Based on the comparative analysis, this paper recommends elliptic curve cryptography algorithm for encryption and decryption of light weight messages in MQTT protocol.

As a result of the analysis done on cryptographic algorithms, it is proved that ECC uses key with smaller size and it is much more efficient than other algorithms. Usage of smaller keys improves the performance of smaller MQTT devices with low power and less storage capacity. So this paper recommends ECC as the best suitable algorithm for key exchange in MQTT protocol and hence leverages asymmetric ECC approach to calculate the common secret key to enable payload encryption in MQTT. Future work includes designing architecture to provide MQTT payload encryption in publisher to subscriber (P2S) and publisher to broker (P2B) scenarios.

ACKNOWLEDGMENT

First and foremost I express my gratitude to the almighty God for his grace. I am extremely thankful to my research supervisor Dr. Raju R Gondkar for his constant support and guidance. I would also like to thank Mr. Samuel M J, Mrs. Sunu George and my family for the encouragement and the immense help given in the completion of this paper.

REFERENCES

1. A. Banks and R. Gupta "MQTT version 3.1.1", OASIS Standard, April 2014.
2. “Implementation of MQTT Protocol on Low Resourced Embedded Network”, International Journal of Pure and Applied Mathematics Volume 116, 2017, 161-166, ISSN: 1314-339.
3. P. T. Eugster, R. Guerraoui, P. A. Felber, and A.M. Kernmarrec, “The many faces of publish/subscribe”, ACM Computing Surveys, vol. 35, no. 2, pp. 114–131, June 2003.

4. OASIS Message Queuing Telemetry Transport Version 3.1.1 http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html#_Toc398718018
5. M. Singh, M. Rajan, V. Shivraj and P. Balamuralidhar, " Secure MQTT for Internet of Things (IoT)", in Communication Systems and Network Technologies (CSNT), Fifth International Conference on IEEE, 2015, pp. 746-751.
6. Diaa Salama Abd Elminaam, Hatem Mohamed Abdual Kader, Mohiy Mohamed Hadhoud, "Evaluating The Performance of Symmetric Encryption Algorithms", International Journal of Network Security, Vol.10, No.3, PP.213–219, May 2010.
7. Mehdi-Laurent Akkar, Christophe Giraud "An Implementation of DES and AES, Secure against Some Attacks", International Workshop on Cryptographic Hardware and Embedded Systems, ISBN 978-3-540-44709-2, September 2001.
8. Kwon, T. (2002). Digital signature algorithm for securing digital identities. Information Processing Letters, 82(5), 247-252.
9. Yuta Hashimoto, Md. Al-Amin Khandaker, Yuta Kodera "An Implementation of ECC with Twisted Montgomery Curveover 32nd Degree Tower Field on Arduino Uno", Volume 8 Issue 2 Pages 341, 2018
10. W.Stallings, Cryptography and Network Security:Principles and Practice, 5th ed. USA: Prentice Hall Press.
11. M. Bafandehkar, S. M. Yasin, R. Mahmod, and Z. M. Hanapi, "Comparison of ECC and RSA algorithm in resource constrained devices", in IT Convergence and Security (ICITCS), International Conference on. IEEE, pp. 1–3, 2013.
12. G R Bamnote, P.R Paradhi, Vivek B Kute "A Software Comparison of RSA and ECC", International Journal Of Computer Science And Applications Vol. 2, No. 1, April / May 2009.

AUTHORS PROFILE

Suja P Mathews is with 14 years of experience into teaching. She has done MCA, M.Phil and is currently pursuing Ph.D in Computer Science from Bharathiar University, Coimbatore. She had worked as Associate Professor in St. Joseph's Arts & Science College and Jyoti Nivas College. Presently she is working for Jain University, Bangalore. Her research work focuses on Internet of Things and the aspects of security enhancements in MQTT protocol. She has published research papers in various International Journals and presented papers in reputed International Conferences.

Dr. Raju R Gondkar has over 22 years of training and teaching experience, and has done extensive research in the areas of Image processing, E-Learning, and Cloud Computing. He contributed significantly in creating the Project report for establishing an Incubation Center with the support of State Government in 2004, to nurture entrepreneurship in IT sector. This, subsequently, became the base for Incubation Centers sanctioned by Government of Karnataka. Incubation centers were also established with IBM and AIT Bangalore under his leadership. He has published and also presented two papers in international conferences – one in Washington DC and the other one in Jamaica, West Indies conducted by IBAM and Common Wealth of Learning. He is currently working as Professor in CMR University, Bangalore.