International Journal of Advance Research in Science and Engineering
Vol. No.6, Special Issue (01), September 2017, BVCNSCS 2017
www.ijarse.com
IJARSE
ISSN 2319 - 8354

Secure Smart Home Using MQTT Protocol An Internet of Things Application

P. Sushma[1], Prof. V. Hara Gopal[2], Prof. M. V. Ramana Murthy[3]
[1]Faculty of Science, [2]Department of Statistics, [3]Department of Mathematics,
Osmania University, Hyderabad (India)

ABSTRACT: *Today, Smart homes is a revolutionary buzz word in the residential space. Smart homes mean a smart living where all the gadgets at home can communicate with each other and can also be operated remotely via a smart phone application. Increasing demand for smart homes are equally increasing the security concerns. A smart device compromising on security can allow hackers to win the control of devices. MQTT (message queuing telemetry transport) is a publish/subscribe based, light weight messaging, easy to use, internet of things protocol. They allow devices to communicate at a faster rate but provide very few security mechanisms. Security in MQTT can be provided in Network layer by using VPN or secure networks, Transport layer by using TLS/SSL and Application layer by using credentials like username and password. This paper aims at providing secure communications using Common division method to encrypt payload in application layer without using full-fledged transport encryption. This is a substitution based block cipher encryption technique, uniquely designed for binary data. This symmetric key encryption technique can be applied on any type of file. Hence it is computationally feasible to apply it on device to device communications. This enhances the security in Smart homes using MQTT by retaining its speed.*

Keywords: Encryption, IOT application, Secure smart homes, MQTT protocol, Common division method, Network security, Decryption.

I. INTRODUCTION

Recent surge in designing the electrical goods or devices to connect to internet and accelerating trend of smart devices flooding residential spaces has laid the ground for the evolution of the smart homes. With the increasing capability to communicate among themselves, the smart devices at home required a greater control. Many applications are developed to control these smart devices remotely. Similarly, many protocols are design for Machine to machine communications. With a stack of protocols available for M2M (machine to machine) communications, MQTT (Message queuing telemetry transport) has took the lead role as it is a simple publish / subscribe protocol with a light weight messaging. It is faster in communications when compared to other protocols. Due to its minimal packet overhead, MQTT is preferred over other protocols and traditional client/server exchanges. It is an "Internet of Things" connectivity protocol. MQTT uses TCP for communications. TCP/IP's port 1883 and 8883 is registered, for using MQTT and MQTT over SSL respectively. These characteristics of MQTT make it ideal for use in constrained environments. Few of the areas where this MQTT is used are Expensive, unreliable, low bandwidth networks and embedded device with limited memory resources or processor.

II. MQTT PROTOCOL

A MQTT Publisher is a client which sends a message to another client and the MQTT Subscriber is a client which receives a message. The message published by a client can be subscribed by another client or the broker can publish it to client. Heart of the publish/subscribe protocol is MQTT broker or Server and it is responsible for authentication and authorization of clients, receiving messages published by clients as topics, filtering them, deciding the interested subscriber, sending the messages to them, holding the session of all persistent clients including subscriptions and missed messages. It is like a central hub through which every message passes. Therefore, the MQTT broker should be able to integrate into backend systems, should be highly scalable, easy to monitor and failure-resistant. Subject based filtering of messages is done by MQTT broker using topics. Topic is a hierarchical structured string or key that identifies the information channel to which the client (Publisher) publishes the payload data. Similarly, the client (Subscriber) uses the topic to identify the information channels on which they want to subscribe the published information.

MQTT uses command messages for communications. These command messages when transferred between client to broker and vice versa, requires secure means of communication. Security from eves dropping and attackers using MQTT can be provided in two layers- the transport layer and the application layer. TLS (Transport Layer Security) and SSL (Secure Sockets Layer) can ensure secure communications but has the overhead of handshaking mechanism between the two devices to establish a connection and to negotiate various parameters. If the connections are used for long durations TLS/SSL serves the purpose but for the smaller messages this significantly increases the CPU usage and communication overhead. So, for any type of communications between the devices we opt for encrypting the payload in application layer.

A. MQTT Quality of Services(QoS)

MQTT uses three QoS's for messages to publish or subscribe. They are as follows:

QoS 0: "At most once", Underlying TCP/IP network makes the best efforts to deliver these messages. Message may be lost or duplicated while using this type of service. for example: ambient sensor data, where a message lost or duplicated does not cause much impact as the next message will be published immediately.

QoS 1: "At least once", here message deliver is assured but sometimes cause duplicates messages to arrive.

QoS 2: "Exactly once", here messages are assured to be delivered exactly once. for example: billing systems, here lost or duplicate messages may lead to incorrect charges to be applied.

MQTT uses a minimal 2 Bytes transport overhead and less protocol exchanges there by reducing the network traffic. Last

Will and Testament feature in this protocol enables interested clients to be notified about the abnormal disconnection.

B. MQTT message format:

MQTT message contains a message header of 2 bytes. Sometimes it could be of variable length too. First byte of the header contains message type and three flags - Duplicate flag, QoS and RETAIN flags. Second byte contains remaining length field. The data values are represented in big endian notation. First part of the header is Message types. There are different message types available for communications. Few of them to mention are as follows:
CONNECT: This is used to connect to the server or broker.
CONNACK: It is a connect acknowledgment.
PUBLISH: Messages sent to other clients, published as topics to broker.
PUBACK: It is a publish acknowledgement.
PUBREC: Publish received is used to intimate assured delivery.
PUBREL: Publish release is also used in assured delivery.
PUBCOMP: Publish complete used in assured delivery.
SUBSCRIBE: Used by client to request for subscription from broker.
SUBACK: Subscribe acknowledgment used as confirm to the subscribers receiving of message.
UNSUBSCRIBE: Used by client to unsubscribe the request.
UNSUBACK: Used to unsubscribe acknowledgement.
PINGREQ: PING request.
PINGRESP: PING Response.
DISCONNECT: Client uses it to disconnect.

MQTT CONNECT message is initiated by client to connect to the server or broker. Broker waits for the stipulated time for the connection to be established but if the client is unable to make a connect the broker or the server rejects the connection. This is done by the broker using CONNACK message to avoid the malicious client from slowing down the broker.

CONNECT message contains the clientID, cleansession, username, password, lastwilltopic, lastwillQos, lastwillMessage, lastwillretain, keepalive fields. Username and password fields are used to authorize and authenticate the client.

MQTT PUBLISH message contains the PacketID, TopicName, QoS, retain flag, payload and duplicate flag fields. This payload is the actual content of the message and this paper aims at encrypting this payload using Common division method.

End to end encryption can be provided using common division method for the MQTT PUBLISH payload and CONNECT payload in application layer. This method works efficiently even in an untrusted environment. Payload of the MQTT publish message is encrypted and the remaining meta data is not considered for encryption. This doesn't require any decryption mechanism at MQTT broker. This enables MQTT Broker to just enroute the payload to subscriber without knowing the content of the payload. Common division method works with binary data and since the MQTT payload is in binary representations, data conversion mechanism is not required.

III. ENCRYPTION USING COMMON DIVISION METHOD

The payload published from publisher to broker and the payload received from broker to subscriber remains encrypted by using common division method. Payload encrypted at publisher will be decrypted at the subscriber only. MQTT broker which can also be a public broker will not be able to read or access the contents of the payload as it is encrypted and at the same time will be able to transfer the payload without any difficultly as the meta data is not encrypted. Key can be accessed to only the trusted clients for decryption. MQTT client can apply this encryption to any topic irrespective of any broker implementation. Even when we don't use TLS services of transport layer for security, this mechanism of payload encryption helps us in securing the data from unauthorized access.

Encryption can be done using two mechanisms: Public and private key encryption, also called as asymmetric encryption and the secret key encryption, also called as symmetric key encryption.

Public and private key encryption uses two keys for encryption and decryption. Public key is used for encryption. And private key is used for decryption. This provides authentication and confidentially to the payload published. The public key is announced to all the other clients but the private key is kept confidentially with the client. Decryption cannot be done with the known public key, client interested in the payload should have the private key for decryption. Only trusted clients can be shared with the private key. This ensures that no other client other than the original user is decrypting the payload.

Secret key encryption uses one key for both the encryption and decryption. "Fig. 1", explains the process of encryption using on MQTT PUBLISH payload using common division method which is a secret key encryption method. This key should be carefully shared between the clients for the future use. Though a common key can be used for all the topics published, it's always advised to use a separate key for each topic. Because a key compromised can cause only one topic to be hacked but if all the topics use the same key, all the topic are having the threat of in secure communications.

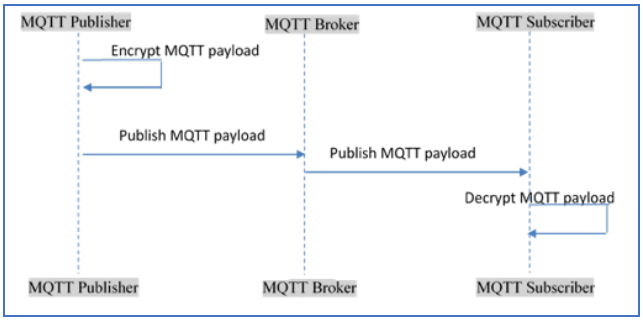

Figure 1. End to end encrytion using common divison method

A. Encryption algorithm:

Let S, Binary stream of data be divided into 'm' number of equal n sized blocks. Last block could be n bits or even less and is not used in encryption process. Let this last block be lb, where 0<=lb<n.

Step 1: Let b1, b2,…bp be ‘p’ consecutive ‘n’ bit blocks such that ‘m’ is divisible by ‘p’.

Step 2: Let ‘d’ be the calculated common divisor of ‘k’ bits maximum(1<=k<=n), of those ‘p’ bit blocks. Let ‘r’ be the integer part of n/p.

Step 3: chose n bit temporary key randomly.

Step 4: Let e1, e2,….ep be temporary ‘p’ number of effective keys generated after ‘r’ bit circular left shift on temporary key.

Step 5: ei and quotient of bi/d are XORed and appended consecutively. Where i=1 to p

Step 6: ‘k’ bit representation of ‘d’ and the results of step 5 are appended to get the encrypted text.

Step 7: next consecutive ‘p’ number of blocks are selected and the above process is continued. The result of this is appended to the previously generated encrypted text, until all ‘m’ number of blocks are involved.

Step 8: Unchanged block ‘lb’ is appended at the front of the concatenated result of step 7. This generates the encrypted text.

B. Decryption algorithm:

The size of each block, the size of temporary key ‘n’, number of consecutive blocks used at a time ‘p’, number of bits to represent the maximum common divisor ‘k’, length of unchanged block ‘n1’ can be obtained from the key.

Step 1: From ‘n’ and ‘p’, the number of bits circularly left shifted can be obtained, Let ‘r’ which produces effective keys e1,e2,e3,……ep from temporary key.

Step 2: ‘lb’ the unchanged block is obtained from the first ‘n1’ number of bits from the encrypted text.

Step 3: Common divisor is produces from the next ‘k’ bits, say ‘d’ of next ‘p’ number of consecutive ‘n’ bit blocks, let them be d1, d2, ….dp.

Step 4: Block ‘i’ is generated by performing XOR operation between ei and di block. Where i=1 to p.

Step 5: Multiply ‘d’ with each of the ‘p’ number of ‘n’ bit blocks example: (block1*d), (block2*d), ……(blockp*d) arrange the resultant product blocks consecutively as blocks in the encrypted text. this generates the first ‘p’ number of ‘n’ bit blocks of decrypted text.

Step 6: Next ‘p’ number of consecutive blocks undergo the same process from step 3 to step 5. Append the next part of the decrypted text with the previous blocks till all the blocks are involved in the process.

Step 7: Original plain text is generated by concatenating the ‘lb’ to the decrypted text.

C. Example: Encyption using Common Division method.

Let the payload of the MQTT PUBLISH message be S = 01101101110111000100101110001101001l. Let us encrypt it using Common division method.

Step1: let us divide this binary stream of data into four 8 bit blocks. So, n=8 and m=4.
Consider p=2, so two consecutive blocks b1, b2 are considered.
Last 4 bits remain unchanged block. And this does not involve in encryption process. So n1=4

Step 2: Let’s calculate the number of bits to represent maximum common divisor d in k. for this payload k=2.
Say r, be the Integer part of n/p. n=8 and p=2, r=4.

Step 3: Generate a random number of length = 8. Let the temporary key be 11011011.

Step 4: Four bits circular left shift generates two effective keys e1, e2.
e1=10111101 and e2=11011011.

Step 5: Perform division operation between bp and d and this result is XORed with effective key ep.

b1/d = 01101101/01=01101101
b2/d = 11011100/01=11011100

(b1/d) XOR e1=11010000
(b2/d) XOR e2=00000111

Step 6: Append the results to get the cipher text of first two blocks of plain text.
It is 011101000000000111

Step 7: Similarly continue with the next two consecutive blocks to get the following binary stream: 11010010011001101

Step 8: The last 4 unchanged bits are appended at the front of two concatenated binary stream.
This gives final encrypted text of the payload as
00110111010000000001111110100100110011011

D. Example: Decryption using Common Division method.

From the key structure we acquire:
Size of each block n=8
Number of consecutive blocks used at a time p=2
Number of bits to represent maximum common divisor (d) k=2
Length of unchanged block (lb) n1=4 and temporary key.

Step 1: From n and p, we get the number of bits circularly left shifted, r=2 and the temporary key to get the effective keys ep. If random number = 11011011, the effective keys e1=10111101, e2=11011011.

Step2: Now consider the first 4 bits from cipher test and store it in lb. so the contents of lb=0011.

Step 3: Then by n and k we acquire the value of d for next p number of n bit blocks and consider (n*p) number of bits from

the rest and differ two 8 bit blocks. The value of d for first two 8 bit blocks is 01 and the blocks are 11010000, 00000111.

Step 4: Next XOR the keys e1,e2 with two 8 bit blocks consecutively. The XOR operation and results are as follows
(11010000) XOR (10111101) = 01101101
(00000111) XOR (11011011) = 11011100

Step 5: The results are individually multiplied by value of 'd'. This is as follows: 01101101*01=01101101
11011100*01=11011100
The resultant block are the first two 8 bit blocks b1, b2 of payload.
It is 0110110111011100

Step 6: This generates only part of payload but the above process should be performed for the remaining chipper text to get the complete payload. Say 111010010011001101, Generate d say 11 for two 8 bit blocks and then XOR the keys with two blocks and then multiply the result by d and get rest two blocks of the payload. Say b3, b4.
b3=01001011 and b4=10001101.
This is 0100101110001101

Step 7: Then append the new results (b3,b4) with (b1,b2). Contents of lb (unchanged block) are appended at the end. This way we can recover the whole payload which is as follows:

01101101110111000100101110001101001 1

IV. CONCLUSION

Common division method encryption and decryption can be resource intensive on constrained devices. It provides secure means of communication between the clients by avoiding unauthorized access and eves dropping. This encryption is well suited when the user can't use Transport layer TLS/SSL and at the same time do not want to transfer the messages in plain text. This method can be further enhanced to apply encryption methods between client and the broker rather than making it end to end. Broker can decrypt the message and send to the subscribers. Client can use both the payload encryption and transport layer TLS encryption in conjunction for higher level of security as payload encryption still has the threat of man in middle attack and message replay.

V. REFERENCES

[1] Bandyopadhyay S.; Bhattacharyya A., "Lightweight Internet protocols for web enablement of sensors using constrained gateway devices," Computing Networking and Communications (ICNC), 2013 International Conference on , vol., no., pp.334,340, 28-31 Jan. 2013.

[2] Colitti, Walter, Kris Steenhaut, and Niccolò De Caro. "Integrating wireless sensor networks with the web." Extending the Internet to Low power and Lossy Networks (IP+ SN 2011) (2011)

[3] Xu Li, Rongxing Lu, Xiaohui Liang and Xuemin (Sherman) Shen, University of Waterloo Jiming Chen, Zhejiang University Xiaodong Lin, University of Ontario Institute of Technology "Smart Community: An Internet of Things Application"

[4] Ullas B S , Anush S , Roopa J , Govinda Raju M "Machine to Machine Communication for Smart Systems using MQTT" International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (An ISO 3297: 2007 Certified Organization) Vol. 3, Issue 3, March 2014.

[5] S.Pandikumar, R.S. Vetrivel 1721 "Internet of Things Based Architecture of Web and Smart Home Interface Using GSM" International Journal of Innovative Research in Science, Engineering and Technology Volume 3, Special Issue 3, March 2014 2014 International Conference on Innovations in Engineering and Technology (ICIET'14) On 21st & 22nd March Organized by K.L.N. College of Engineering, Madurai, Tamil Nadu, India M.R. Thansekhar and N. Balaji: ICIET'14

[6] Ming Wang; Guiqing Zhang; Chenghui Zhang; Jianbin Zhang; Chengdong Li, "An IoT-based appliance control system for smart homes," Intelligent Control and Information Processing (ICICIP), 2013 Fourth International Conference on , vol., no., pp.744,747, 9-11 June 2013.

[7] Min Chen, Jiafu Wan,Gonzalez S., Xiaofei Liao, Leung, V.C.M., "A Survey of Recent Developments in Home M2M Networks," Communications Surveys & Tutorials, IEEE , vol.16, no.1, pp.98,114, First Quarter 2014.

[8] Karia, D.C., Adajania, V., Agrawal M., Dandekar S., "Embedded web server application based automation and monitoring system," Signal Processing Communication Computing and Networking Technologies (ICSCCN), 2011 International Conference on , vol., no., pp.634,637, 21-22 July 2011

[9] Sujoy Dasgupta, Sanjit Mazumder, Prof. (Dr) Pranam Paul "Implementation of Information Security based on Common Division" IJCSNS, VOL.11 No.2, February 2011

[10] Mandal J. K., Mal S., Dutta S., "A 256 Bit Recursive Pair Parity Encoder for Encryption", accepted for publication in AMSE Journal, France, 2003

[11] Dutta S., Mal S., "A Multiplexing Triangular Encryption Technique – A move towards enhancing security in ECommerce", Proceedings of IT Conference (organized by Computer Association of Nepal), 26 and 27 January, 2002, BICC, Kathmandu.