An advance encryption standard cryptosystem in iot transaction

Zolidah Kasiran, Shapina Abdullah, Normazlie Mohd Nor
Faculty of Computer & Mathematical Sciences, Universiti Teknologi MARA, Malaysia

Article Info

Article history:

Received Jun 14, 2019
Revised Sep 15, 2019
Accepted Sep 29, 2019

Keywords:

Advance encryption standard
Internet of things
IoT security
Secure transaction

ABSTRACT

IoT create an ecosystem that can be useful to the world with its various services. That make the security of IoT is more important. This paper presents a proposed technique to secure data transaction from IoT device to other node using cryptography technique. AES cryptography and IoT device model ESP8266 is used as a client to send data to a server via HTTP protocol. Security performance matrices used eavesdropping attack with Wireshark sniffing and brute-force attack. will be simulated to the proposed method in order to ensure if there is any possibility it can be cracked using those attacks and to evaluate the performance of ESP8266, several experiments were be conducted. Result gather from this experiment was evaluated based on processing time to see the effectiveness of the platform compare to different data size used while adapting this technique.

Corresponding Author:

Zolidah Kasiran,
Faculty of Computer & Mathematical Sciences,
Universiti Teknologi MARA,
Shah Alam, Malaysia.
Email: zolidah@tmsk.uitm.edu.my

1. INTRODUCTION

The IoT vision is to revolutionize the Internet, to create networks of billions of wireless identifiable objects and devices, communicating with each other anytime, anyplace, with anything and anyone using any services [1]. It is foreseen in the near future, everything from individuals, groups, objects, products, data and services will be connected and impacted by the IoT paradigm [2]. The rapid development of application domains and innovative services are expected that may change the way people live, work and communicate[2]. By year 2020, IoT ecosystem is expected to reach 5.8 billion connected devices [3].

With the increase number of devices connected to the IoT ecosystem, the security and privacy issues are observed and it became more pronounced [1]. As IoT devices supposed to make their data accessible to interested party, doing it in controlled way has become the major issue [4]. Data transmitted may contain sensitive information about users' private lives, habits, activities and relations, which all refer to individuals' privacy [3, 5]. It is therefore crucial that IoT systems must guarantee the confidentiality and integrity of the information and the privacy and anonymity of users. Readable data transmission is compromised being altered throughout the process. Implementing cryptosystem into data security to increase confidentiality is needed. All transmitted data will be in encrypted format during transmission where they are in cipher text mode [6-7].

IoT application is vary and can be used in a simple application like home automation to the complex energy conservation commercial and domestic demand management [8]. Healthcare is one of the area that taking advantage of IoT technologies such as heart beat rate monitoring, as body temperature, blood glucose level, and blood pressure [9-10]. Wearable body sensor networks (WBSN) is used to continuously monitor patient activities or medical parameters [11]. Furthermore, IoT devices also could be used to monitor patient's current medicine and evaluate the new medicine in terms of allergic to the medicine [12].

1.1. Internet of Things Security

As the technologies of IoT are advancing, the number of devices connected to each other are also increasing. These devices provide an unprecedented ability to sense and control the environments. Sensor data can be transmitting between devices by remote data to the centre to be analysed. Security for this technology is still in the big question because it still unsecured in nature. For the embedded systems it is not a new issue for security [13]. Security issues such as eavesdropping can illegitimately snoop into people's daily activities, even a legitimate entity may be gathering data without the user's consent. The simplest way of ensuring data secrecy is to apply encryption to the data.

1.2. Application Layer Security

This section discusses about security that applied at application layer such as, Rivest-Shamir-Adleman (RSA), Advance Encryption Standard (AES) and Elliptic Curve Cryptography (ECC) cryptosystems. These three symmetric cryptosystems are the common technique applied by researcher in this field.

1.3. Rivest-Shamir-Adleman (RSA)

The RSA cryptosystems have been used widely in securing data transmission [14]. The variant of RSA research has been done by [15-16]. T heir outcome of the research is to improve processing speed and also to have alternative ways to implement RSA cryptography. Another research by [17] used RSA prime number factoring implemented in MATLAB. The research had shown that, the key generation and encryption/decryption process was taking less time compared to normal RSA prime number factoring. Other researchers [18-19] had implement RSA cryptosystems on Arduino IoT platform and they use additional device such external memory and Java Card to help Arduino to do the key generation and processing (encryption / decryption process). Different results were observed from both researchers.

1.4. Elliptic Curve Cryptography (ECC)

The use of IoT device in healthcare environment has invited the intention to secure communication between IoT device and also data privacy. Researcher [20] has applied Elliptic Curve Cryptography (ECC) to the RFID devices that used in healthcare environment. A new RFID authentication has been proposed alongside with ECC implementation. The performance shows that the proposed protocol works better and more suitable to be applied in healthcare environment. In Wireless Body Area Network (WBAN), [21] had applied ECC cryptosystems the healthcare application. The proposed technique that uses ECC as their security module has resolved the problem that is replay attack and mutual authentication. It also proves that it can operate on the IoT device with resource constrained.

1.5. Advance Encryption Standard (AES)

The Advance Encryption Standard (AES) is a strong encryption algorithm which has several advantages in data ciphering [22]. However, AES also have disadvantages such as high computations, and hardware requirement. These problems become more complicated when AES algorithm is used to encrypt the images data especially high definition format images. A modified AES proposed by [23] has less computational intensive and compatible with high definition images.

Test on AES performance experiment has been done by [24] in terms of different AES implementation. There are three different implementation used in this experiment that AES standard, AES fast implementation and AES-NI extended. The result of the experiment shows that AES-NI extended is the fastest among them with the help of hardware. Secondly, AES fast implementation algorithm is the optimized version of AES realized by software, which also very fast compared to AES standard. More research by [25] had implemented AES securing data in cloud environment.

2. RESEARCH METHOD

This section discusses the method and process for this experimental of encryption technique and the flow of the process in detail. It consists of several process and technique to produce desired output. The main idea is to encrypt all transaction throughout client (ESP8266) and server, and to complement proposed encryption technique, base64 encoding/decoding technique is used to make the encrypted transaction more secure. The processes that were involved in this research are listed as below.

a. Encrypt/Decrypt using AES cryptosystem with AES-128-cbc mode.
b. Encode/Decode the encrypted data using base64.

The proposed technique involved two side operations and each side has its own processes. Process start by client at first gets the credential which is its chip ID. This credential was encrypted and encoded before it sends out to server to initiate the session. The Server processed the credential by decode-decrypt and verify it and if it valid, an authorization is granted to the client by sending a reply code. Replied code then gone through a decode-decrypt process before it can be verified with authentication code. The verified code had enable the client to start to read the current temperature data from DS18B20 temperature sensor. This data later was encrypted using AES-128 before they encode using base64 before send out to server. The client keep reading and sending the temperature data with two minutes frequency until connection between client and server go offline.

The steps of the process of Encrypt and Decrypt Credentials is shown in Figure 1. The process requires authenticating the connection between ESP8266 and server. An ESP8266 chip's ID is used as the unique key for the credential. The key is read and chunked into several strings to reduce the time of the encryption and encoding process. The strings is sent to the server after all chunk had been encrypted and encoded. The strings received by server is decoded and decrypted and known as credential between ESP8266 and server as the link authentication.

```
Process 1. Encrypt-Encode credential
Input: Chip Identification
Output: encoded encrypted Chip Identification
1. get Chip Identification from IoT device
2. chunk the data
3. encrypt all chunked data
4. encode encrypted data
5. send encoded data to server
```

Figure 1. Encrypt and encode chip identification process

Once credential successfully received by the server, a chipper text is generated and sent to the client as a reply code. A Decode and Encode process is need to be performed for key validation. Figure 2 shows the Decode-Decrypt process.

```
Process 2. Decode-Decrypt data
Input: Incoming data (chiper text reply code) from server
Output: original version data
1. get incoming data from server
2. decode the data
3. decrypt the data
4. validate the decrypted data
```

Figure 2. Processes for decode and decrypt process

The Figure 3 shows the step by step activities to encrypt and encode temperature data gathered from temperature sensor. This process occurs after the return value from server validated and match with the value set in client memory. Encoded temperature data then send to the server.

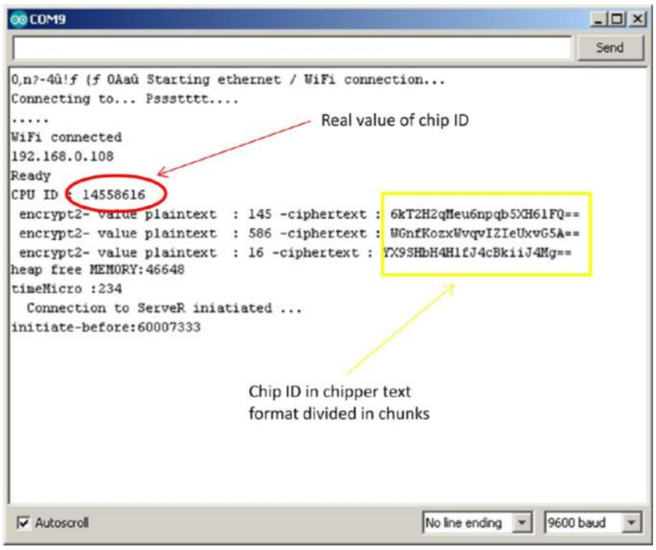
```
Process 3. Encrypt-Encode data
Input: data (temperature)
Output: encoded version of data
1. get data
2. chunk the data into 2 chunks
3. encrypt both chunk data
4. encode both encrypted data
5. combine encoded data1 & encoded data2 into a string parameter
6. send encoded string to server
```

Figure 3. Processes for encrypt and encode temperature data

3. RESULTS AND ANALYSIS

In this section the performance of proposed encoding-decoding AES technique is evaluated. All the result was captured and compared to measure the effectiveness of this proposed technique while adapting into the IoT device. Eavesdrop and brute force attacks were simulated in order to test the data confidentiality and processing time of encryption and decryption process was used to measure the performance of the IOT device. Considering that all the experiment done using the same development platform, all results gather from the experiment summarized or plotted into graph so the effectiveness and performance of the technique could easily understandable.

The Figure 4 show the information of credential and data before they send to the server of ESP8266. Figure. 4 shows the credential information, it shows the IOT chip ID real value before it chunked into three portions and going through encrypt-encode process. As a result, all chunks transformed to chipper text version.



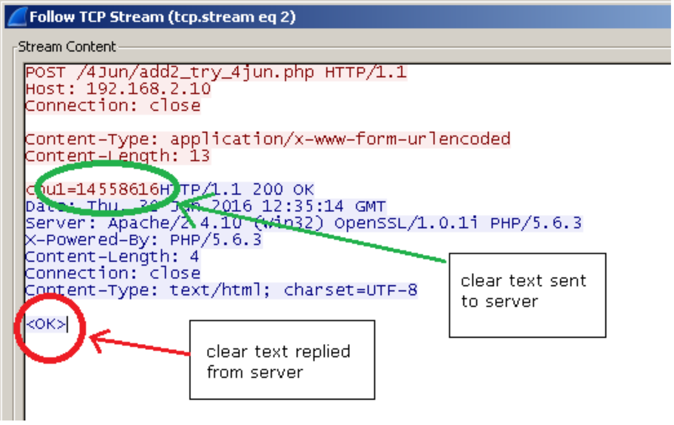
```
0,n?-4û!ƒ (ƒ 0Aaû Starting ethernet / WiFi connection...
Connecting to... Psssttttt....
.....
WiFi connected
192.168.0.108
Ready
CPU ID : 14558616
 encrypt2- value plaintext  : 145 -ciphertext : 6kT2H2qMeu6npqb5XH61FQ==
 encrypt2- value plaintext  : 586 -ciphertext : WGnfKozxWvqvIZIeUxvG5A==
 encrypt2- value plaintext  : 16 -ciphertext : YX9SHbH4H1fJ4cBkiiJ4Mg==
heap free MEMORY:46648
timeMicro :234
  Connection to ServeR iniatiated ...
initiate-before:60007333
```

Real value of chip ID

Chip ID in chipper text format divided in chunks

Figure 4. Sample output from ESP8266 For Encrypt-Encode Of Chip ID

The Figure 5 shows the data packets before the proposed technique is applied to the application were captured and analyzed. Transaction of sending and receiving reply data are done in clear text, so it can be read and understandable. As proofed above, an eavesdropper can easily get the credential which is important to authenticate the connection and can use it to manipulate the data for their own purposes.



```
POST /4Jun/add2_try_4jun.php HTTP/1.1
Host: 192.168.2.10
Connection: close

Content-Type: application/x-www-form-urlencoded
Content-Length: 13

cpu1=14558616HTTP/1.1 200 OK
Date: Thu, 30 Jun 2016 12:35:14 GMT
Server: Apache/2.4.10 (Win32) OpenSSL/1.0.1i PHP/5.6.3
X-Powered-By: PHP/5.6.3
Content-Length: 4
Connection: close
Content-Type: text/html; charset=UTF-8

<OK>
```

clear text sent to server

clear text replied from server

Figure 5. Packets captured before encryption applied

3.1. Impact on Processing Time

The following section discuss about the result on the time impact for both encryption and decryption on different sizes files.

3.2. Encryption

The Encryption is a process where the original data is converted into an encrypted form based on the shared-key. Processing time for every encryption depends on how big data size that need to be processed. Table 2 has recorded time taken for every data size involve in this application.

The result from the Table 1, shows that the different processing time due to different data size for encryption process. For 4 bytes data size, the time taken is 17289 microseconds which is very fast to complete the encryption cycle. For 10 bytes of data size, and processing time for this data size is 38036 microseconds. Compared to 4 bytes of data size, 10 bytes data size takes 20747 microseconds more time to finish the encryption process. For 50 bytes of data size, the processing time is increase to 180179 microseconds. Compared to 10 bytes of data size, the processing time gap is 142143 microseconds. For 80 bytes of data size, processing time taken to complete the encryption cycle is 285268 microseconds. The time gap between 50 bytes and 80 bytes data size is 105089 microseconds.

As shown in the result we can agree that data size of transaction will impact the processing time. It simply requires longer time to complete the encryption process when the data size increased.

Table 1. Time Characteristic of Encryption Process

Data Size	4 bytes	10 bytes	50 bytes	80 bytes
Time (μs)	17289	38036	180179	285268

3.3. Decryption

The decryption is a process where the encrypted data is converted back into its original form using the shared-key. Processing time for every decryption process depends on how big data size that need to be processed. Table 2 has recorded time taken for every data size involve in this routine.

Table 2. Time Characteristic of Decryption Process

Data Size	4 bytes	10 bytes	50 bytes	80 bytes
Time (μs)	14686	32309	126006	189008

Result shown in the Table 2 had shown that the different processing time due to different data size for encryption process. For 4 bytes data size, the time taken is only takes 14686 microseconds to complete the routine. Meanwhile 17623 microseconds is the additional time needed to decrypt 10 bytes of data compare to 4 bytes data. Furthermore, significance increment of processing time required to decrypt 50 bytes of data size, the processing time is 126006 microseconds. Lastly, 80 bytes of data size, processing time taken to complete the encryption cycle is 189008 microseconds. The time gap between 50 bytes and 80 bytes data size is 63002 microseconds.

As shown in the result, decryption process an average score a shortest completion time compared to encryption process and same as encryption, data size of transaction will impact the processing time.

The experiment had shown that cryptography can help securing the data transmission of IoT. As result of the experiment had shown that intercepted data cannot be read and without the exact key all intercepted data are useless. Cryptography is the way to change data into human unreadable form and it works in pair, encrypt-decrypt and encode-decode. And by adopting AES-128 security algorithm into the IOT transaction could prevent data privacy from eavesdropping and brute-force attacks. Since AES is a symmetric cryptography, to keep the shared-key secrecy is very important.

In term of performance, AES-128 performs reasonably well on ESP8266 platform. Based on experiments done, the results show different processing time for different process. It happens due to the different activities that happen in the different process. As for encryption and decryption process for example, both process going through the same activities but in different order. Thus, the time different between these processes are small and the time gap approximately parallel. It is the same as what happen to encode and decode process, even though each of the process go through the same activities, but the route taken to complete the activities are different.

By simulating the data size transmitted to the server, it proved data size significantly impacted the performance of the IoT device. Longer processing time need to perform every cryptography process when data transmit is bigger.

4. CONCLUSION

This project is about security on IoT in a small device. The idea is to transform data sent from client to its server through HTTP protocol into chipper text mode. AES-128 was proposed since it is a very reliable cryptography technique. A test bed has been developed and there are three processes involve in the securing the data communication, which are encrypt-encode for credential, decode-decrypt reply from server and encrypt-encode for data. For credential process, it happens only once in a system lifecycle, where after the process end, encoded credential is save for connection initiation with server. To ensure this proposed technique would secure IoT data transaction, several experiments was performed, Wireshark sniffing tool is being used to capture the transmitting data, and as expected data captured cannot be read because they are in chipper text mode. Then brute-force attack simulation is performed to AES-128 key to find out the possibility whether it can be cracked. The result shows it is almost impossible to crack AES-128 key combination by considering the time taken and resource required to perform the activity, and for the performance of ESP8266 on adopting this method, processing times are evaluating as the bench mark at every test performed. As the result show and proofed above, data size significantly impacts the performance. The bigger data size transfer means bigger processing time require.

ACKNOWLEDGEMENTS

This research was financially supported by the Ministry of Higher Education through Fundamental Research Grant Scheme 600-RMI/FRGS 5/3(0006/2016).

REFERENCES

[1] Abomhara M, Geir M. Køien. "*Security and privacy in the Internet of Things: Current status and open issues".* International Conference on Privacy and Security in Mobile Systems (PRISMS), 2014.

[2] Chatzigiannakis, I., *et al*, "A privacy-preserving smart parking system using an IoT elliptic curve based security platform". *Journal of Computer Communications. 89*(90), 165-177, 2016.

[3] Price, B. A., Adam, K., & Nuseibeh, B (2005). "Keeping ubiquitous computing to yourself: a practical model for user control of privacy". *International Journal of Human-Computer Studies, 63*(1-2), 228-253, 2005.

[4] Hossain, M, *et. al* (2015). "Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things". *IEEE World Congress on Services*, 2015.

[5] M Yusof M A; M Ali F H; & Darus M Y, "Detection and Defense Algorithms of Different Types of DDoS Attacks*", International Journal of Engineering and Technology, 9(5*), 2017.

[6] Hussein S M; Mohd Ali F. H ; Kasiran Z, "*Evaluation effectiveness of hybrid IDS using Snort with Naïve Bayes to detect attacks*", Second International Conference on Digital Information and Communication Technology and it's Applications (DICTAP), 2012.

[7] Kasiran Z, Napi R (2018),"Randomize IPv6 Stateless Address Autoconfiguration in None-stable Storage Arduino Devices*", Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, 12 (1), 254-260,2018.

[8] Smith, D. R., & Smith, R. B. "*Innovative Solutions for Energy Conservation Through Commercial and Domestic Demand Side Management*". Energy Procedia,*2015; 79,* 169-174, 2014.

[9] Chooruang, K., & Mangkalakeeree, P. "*Wireless Heart Rate Monitoring System Using MQTT*". Procedia Computer Science, *86*, 160-163, 2016.

[10] Xu, B., et.al.."Ubiquitous data accessing method in IoT-based information system for emergency medical services". *IEEE Transactions on Industrial Informatics*, 2014.

[11] Miorandi, D., *et al*. "Internet of things: vision, applications and research challenges". *Ad hoc Networks*, 10(7), 1497-1516, 2012.

[12] Jara, A. J.,et.al. "*Pharmaceutical Intelligent Information System to Detect Allergies and Adverse Drugs Reactions based on Internet of Things*". 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010.

[13] Ukil, A., Sen, J., & Koilakonda, "*Embedded security for Internet of Things".* 2nd National Conference on the Emerging Trends and Applications in Computer Science (NCETACS), 2011.

[14] Athinarayanan S, Nivetha P, Supriya R. "Secure Data with Key Managers by Using Shamir Scheme and AES Algorithm*". International Journal of Computer Science Trends and Technology*,5(2), 298-301, 2017.

[15] Balasubramanian, K. (2014)."*Variants of RSA and their cryptanalysis";* International Conference on Communication and Network Technologies (ICCNT),2014.

[16] Zhao, G., & Li, H. "*An Efficient Variant of the Batch RSA Cryptosystem*", International Conference on Communication, Electronics and Automation Engineering (pp. 947-954). Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[17] Wang, Het.al. "*Key generation research of RSA public cryptosystem and Matlab implement*". International Conference on Sensor Network Security Technology and Privacy Communication System (SNS & PCS), 2013.

[18] Al-Haija, Q. A., et.al. "*A Tiny RSA Cryptosystem based on Arduino Microcontroller Useful for Small Scale Networks*". Procedia Computer Science*, 34,* 639-646, 2014.

[19] Zhang E. P, et.al, "*A Simple and Efficient Way to Combine Microcontrollers with RSA Cryptography*". The World Congress on Engineering and Computer Science*, Vol I*, 5, 2013.

[20] Zhao, Z. "A Secure RFID Authentication Protocol for Healthcare Environments Using Elliptic Curve Cryptosystem". *Journal of Medical Systems,* 38(5), 1-7, 2014.

[21] Mathew, J., et.al. "*Systems Secure Medical Data Transmission by Using ECC with Mutual Authentication in WSNs*". Procedia Computer Science*, 70,* 455-461,2015.

[22] Shraddha D. "Performance Analysis of AES and DES Cryptographic Algorithm on Windows and Ubuntu Using Java". *International Journal of Computer Trends and Technology* 35(4), 179-183, 2016.

[23] Salim, J., et.al. "*Rapid Encryption Method based on AES Algorithm for Grey Scale HD Image Encryption*". Procedia Technology, 11, 51-56, 2013.

[24] Guo, G. l., Qian, Q., & Zhang, R. "Different Implementations of AES Cryptographic Algorithm*". International Symposium on Cyberspace Safety and Security (CSS).*2015.

[25] Thalari B T, Vootla H, Priyanka K (2017). "Encryption and Decryption- Data Security for Cloud Computing Using AES Algorithm.". *International Journal of Computer Trends and Technology*,2017.