Internet of Things Data Integrity

Gift MATSEMELA, Suvendi RIMER, Khmaies OUAHADA, Richard NDJIONGUE, Zinhle MNGOMEZULU

Department of Electrical and Electronic Engineering Science, University of Johannesburg, P.O Box 524, Auckland Park Kingsway Campus, Johannesburg, 2000, South Africa
Tel: +27 11 559 2147, Fax: +27 11 559 2344, Email: kouahada@uj.ac.za

Abstract: Internet of Things is an internetwork where people interact with the environment via sensors and machines to perform physical activities remotely. IoT allows objects to be sensed and controlled remotely via internet such as changing the room temperature by controlling an air conditioner, switching on the lights in your apartment, using smart watches to gather information about your body health (blood pressure and heart beat rate) and sends it to your private doctor to monitor it and so on. Performing a research on issues faced with IoT, it was realized that most challenges were of privacy/confidentiality and data integrity. This project further conducts research on finding ways of securing information that is shared between two or more interactive IoT devices. The main objective of the project was to develop a strategy of securing the IoT technology world. The method that was proposed to provide data security and integrity was through cryptographic methods/security algorithms that require decent memory and equitable CPU processing power .There were three security methods that were focused on in this project i.e. RSA, AES and TDES algorithms. Each of the encryption/decryption methods were presented with their detailed design and functionality. The most suitable and better performing security algorithm for IoT technology was chosen according to the measures of time, memory and processing power. With the tests performed on the proposed security algorithms, AES encryption standard seemed to perform better than other security algorithms in terms computational time, memory consumption and the required processing power. The test results also showed that it has a large encryption/decryption key size which also means better protection for data. However through testing of the encryption methods there were shortcomings that were encountered such as inability of the security algorithms to provide authentication and encrypt different data formats such as video, picture and audio. The short comings will be further dealt with on the future work. The success of this research has potential impact on increasing the security of IoT internetwork which intern will increase the number of users in the IoT world.

Keywords: IoT, RSA, AES, TDES, cryptography, security algorithms

1 Introduction

Internet of Things is a network of physical objects such as devices, vehicles, buildings and machines lodged with electronic circuits, software, sensors and network connectivity that enables these objects to communicate, collect and exchange data with their internal states and external environment [1]. It is basically about connecting devices to the internet, letting them talk to us through applications. The IoT technology allows objects to be sensed and controlled remotely via the internet such as controlling the room temperature, switching on the lights at your apartment and so on. The practical recognition of IoT requires the occurrence of a different number of platforms and technologies, this is with regard to the process of identification and tracking, sensing, communication, computational sensing, coordinated and distributed control, semantic knowledge processing, traffic, and user modelling [2].

The IoT technology is been broadly applied in intelligent building, smart televisions, public security, smart watches and so on. Because of its great market prospects, IoT has been given a full close

attention by several governments all over the world, which is now rated as the third wave of information technology after the internet and mobile communication network [3].

IoT security is about securing data that has been acquired and exchanged between IoT devices. It is required that the data exchanged between these IoT devices is immensely secured and the devices themselves are protected. This is because the data shared between devices is very confidential and important as it contains personal credentials, health status, financial statements and other confidential stuff. Thousands of IoT-connected devices deliver new experiences to people throughout the world and lowering costs [4]. Unfortunately, this growth of connected devices brings increased security risks and threats evolve quickly. Serious risks include physical harm to people (this includes hacked smart locks and bridged baby monitors) and damaged industrial equipment such as pipelines, blast furnaces, and power generation facilities [5]. As several such facilities and IoT systems have already been attacked and materially damaged, security is now an essential consideration for everyone operating an IoT devices or systems.

2 Objective

The main objective of the project was to develop a strategy of securing IoT technology using security algorithms that require decent memory and equitable CPU processing power. This is done with an aim to improve the security of IoT technology and protecting user's confidential information. The security algorithms should however be able to:

- Encrypt/decrypt data.
- Have a manageable key size.
- Compatible with all IoT devices
- Provide confidentiality and data integrity to users

3 Methodology

The methodology of the security algorithms was based on cryptographic methods that are currently existing. With various cryptographic methods existing and utilized in different sectors such as banks biometrics and others, for this project only three alternative cryptographic methods were selected. The selection was based on the project's requirement. The selected cryptographic methods were RSA, AES and TDES. Each method is briefly defined and experimented in order to find the most efficient security algorithm. The description of each security algorithm is as follows:

3.1 Triple DES

Triple DES (TDES) was created back in the years when the original DES was becoming weaker. The motive of designing triple DES was to offer users with more security strength in an easier way and accommodating systems that were already using the old DES, as it would be much easier using multiple DESes composite function than bolting new ciphers [6]. TDES also known as Triple Data Encryption Algorithm is a symmetric key cryptosystem where both the sender and receiver use the same key to perform the encryption/decryption process. TDES provides security by utilising three cipher blocks and three keys in every information block [6]. TDES algorithms can be fairly operated on low level computers such as micro-controllers with a minimum CPU of 12 MHz and a memory of 128-500MB (RAM) [7]. TDES algorithm can be developed in C and python language. Triple DES is prone to attacks such as meet-in-the-middle, which is an attack based on cipher blocks [7] i.e. it utilises cryptographic functions of each cipher block to bypass the protected message. Meet-me-in-the-middle attack also reduces the required brute force permutations required to break the protected data that uses similar keys to secure confidential information. Figure 3.1 below depicts the operation of TDES.

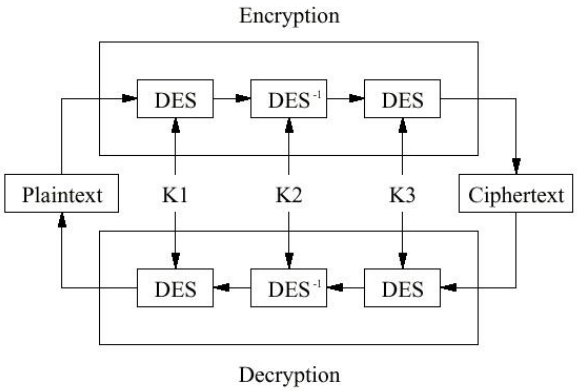

Figure 3.1: Depictions of TDES security algorithm

3.2 AES

AES encryption/decryption standard is also a symmetric cryptosystem. It is type of cryptosystem that was built to be efficient in both software and hardware [8]. It is an iterative encryption/decryption technique based on a mathematical rule of substitution-permutation. The AES encryption/decryption process setup includes a series of connected operations, in which some of the operations deal with replacing inputs with specific outputs and other operations involve rearranging bits around. AES cryptosystem uses rounds to secure information [8]. For 128 bits key it uses 10 rounds, for 192 bits key it uses 12 rounds and for 256 bits key it uses 14 rounds. All keys are considered adequate enough to secure classified data, although for top secret information 192 and 256 key lengths are considered to be appropriate. AES performs all its computations on bytes instead of bits i.e. it treats 128 bits of plaintext as 16 bytes. These 16 bytes are organized in columns and rows for processing of a matrix. AES algorithms are able to operate on low level computers that have a CPU of 12MHz and a minimum memory of 126MB [8]. Unlike triple DES, AES is prone to side-channel attack, which is an attack that focuses on utilising information gained on the physical application of the cryptographic method rather than brute force or weakness in the algorithm. AES algorithms also have challenges with key management, as the key for both the sender and receiver has to be the same, this results in sharing the key via internet which is still not secure [9]. Figure 3.2 below shows he operation AES algorithm.

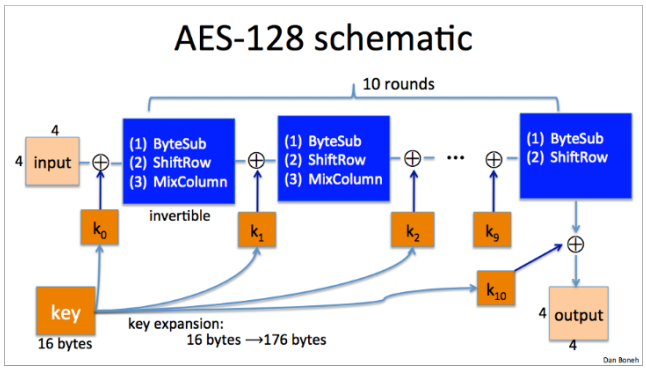

Figure 3.2: Depiction of AES security algorithm

3.3 RSA

RSA is a cryptographic standard that was founded by Ron Rivest, Adi Shamir and Leonard Adleman. It is the first encryption standard that uses public key cryptosystem and it's applied in modern computers to secure electronic data transfer [10]. The encryption standard is based on asymmetric key i.e. the sender and receiver use two different keys. However the keys are mathematically linked, the published key by the sender is publicly displayed to everyone who wants it, the other key is private

and it is only known by the receiver. RSA cryptosystem uses key size of 1024 to 3072 bits. Data encrypted by public key can only be decrypted by a specific private key. This type of cryptography usually requires authenticity to ensure that the published public key mathematically matches with the intended private key. The strength of RSA cryptography depends on the key generation and encryption functions [10].

Unlike other encryption methods RSA does not specifically operate on strings of bits the same way as symmetric key encryption standards, it works on numbers [11]. It's basically about factoring large integers that are product of two large prime numbers and finding modulus of the resulted product. For better security in RSA cryptosystem, it is important that the plaintext is represented as series of numbers less than modulus N [11]. This type of cryptosystem is mostly used in electronic communication systems and data storages as it assures confidentiality, integrity and authenticity. Even though RSA cryptosystem seem to have a very strong key length, however it is vulnerable to some attacks such as timing attacks [11]. Timing attacks are attacks that based on acquiring information on the implementation of the cryptosystem rather than the weakness in the mathematics of the cryptosystem. Timing attacks utilize the time variations in the encryption/decryption process and use it to compute the required private key.

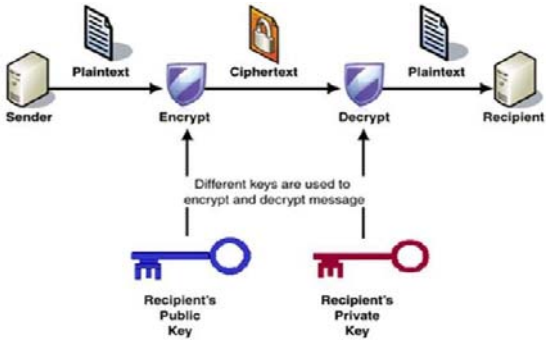

Figure 3.3: Operation of RSA algorithm

4 Technology Description

With the three proposed security algorithms only two algorithms were meeting the project's requirements i.e. RSA and AES security algorithms, thus the two security algorithms were compared for this project. The technological description and detailed designs of the security algorithms are as follows:

4.1 AES

AES is a subgroup of Rijndael Cipher that can process a block cipher of 128 bits (block length). AES algorithms allow cipher keys of 128 (AES-128), 192 (AES-192) and 256 bits (AES-256). The algorithm uses rounds to perform the encryption/decryption process and the number of rounds are dependent on the size of the key. For AES-128 it utilizes 10 rounds, for AES-192 it utilizes 12 rounds and AES-256 it utilizes 14 rounds. The following table summarises the AES security levels and their keys i.e. key-block-round combination. On table 1, NK represents the number of 32 bits making up the cipher key, Nr represents the number of rounds corresponding to each type of AES and Nb represents number of columns making up the state.

The basic unit of operation that AES algorithms use is a byte [12]. The input plaintext, output ciphered text and encryption key are arranged and processed in an array of bytes which is formed from a subset of 8 bits. These bytes can be seen as finite field elements using hexadecimal or polynomial representation. Inside AES algorithms, the operations are performed in a state i.e. a 2-dimensional array of bytes [12]. Inside the State array (represented by the *S)*, each byte in the array has two pointers(index), one pointer represents row number r in the range of $0 \leq r < 4$ and the other pointer represents column number c in the range of $0 \leq c < \boldsymbol{Nb}$. This allows each byte of the State to be assigned as either Sr,c or $S[r,c]$. At beginning of the encryption process, the input text is copied

into the state array. The state array is then transformed using four transformations. The four transformations are substitution, shifting, mixing and round key. The substitution transformation is responsible for substituting the bytes in the S-box, whiles the shifting transformation is responsible for shifting rows of the state array, mixing transformation is responsible for mixing the columns of the state array and lastly the round key transformation is responsible for adding a round key at end of a round [12].

4.2 RSA

RSA cryptosystem was designed as result of the problem with key distribution in other existing cryptosystem [13]. RSA cryptosystem is a type of cryptosystem that eliminates the need of using messengers to deliver keys to recipients over a different secure channel before sending the original intended message [13]. In RSA cryptosystem, the encrypting key is public whiles the decrypting key is private, this means that for decryption only the person with the matching decryption key can decrypt the ciphered message. The public key and private key are mathematically related, however they must be made in such a way that they must not be easily deduced from each other.

RSA cryptosystem uses numbers as the base unit for performing encryption. It is based on three theorems, which are as follows:

Theorem 1: Fermat's Little Theorem states [14]***;***

If p *is a prime number, and is an integer such that* (a, p) = 1, *then* $a^{p-1} = 1 (mod\ p)$.

Theorem 2: Fermat's Theorem Extension states [14]***;***

If (a, m) = 1 *then* $a^{\Phi(m)} = 1 (mod\ m)$, *where* Φ (m) *is the number of integers less than* m *that are relatively prime to* m*. The number* m *is not necessarily prime.*

Theorem 3: Chinese Remainder Theorem states [14]***;***

Let p and q be two numbers, but such that (p, q) = 1. Then if a = b (mod p) and a = b (mod q) we have a = b (mod p q).

In RSA security algorithms, there are four steps that are performed, which are the key generation, key distribution, encryption and decryption [14]. For key generation this includes making a public key and developing a matching private key. For key distribution this includes transmitting the public key to the recipients via a reliable route, the private key is never distributed. For encryption and decryption this includes converting a plaintext to numbers and numbers back to plaintext.

5 Developments

In order to know which security algorithm meets the project's requirements, experimental test were carried out. For this project three essential experiments were performed i.e. functionality of security algorithms, Speed of encryption/decryption and CPU usage of the security algorithms. The procedure for each experimental test are as follows:

5.1 Functionality of the security algorithm

5.1.1 Procedure for AES algorithm

- From AES setup, execute the algorithm
- Choose the key length between 128,192 and 256.
- Enter your preferred key in hexadecimal.
- Enter the plaintext desired to be protected and run the algorithm

5.1.2 Procedure for RSA algorithm

- From the RSA setup, compile and execute the algorithm
- Enter the first prime key number, then the second prime key number
- Enter the message desired to be protected and there after run the code

5.2 Speed of encryption/decryption

5.2.1 Procedure for AES algorithm

- Run algorithm setup for AES
- Choose the minimum key length

- Enter the encryption key
- Enter the message and execute the whole code
- Then measure time and the amount of data to be encrypted

5.2.2 Procedure for RSA algorithm

- Run the setup key for RSA algorithm
- Choose and enter two large prime numbers as your key
- Enter the message and execute the whole security code
- Measure time and the amount of data to be encrypted

5.3 CPU usage of the security algorithm

5.3.1 Procedure for AES algorithm on an Arduino

- Connect the Arduino to the computer via Arduino cable.
- Set the com number where the Arduino is connected.
- Compile the algorithm
- Upload the algorithm, enter the key, enter message and run it.
- Measure the processing power

5.3.2 Procedure for AES algorithm in Raspberry pi

- Setup the Raspberry pi using relevant components such HDMI, power supply adapter and memory card.
- Run the algorithm, enter the encryption key and the message
- Measure the processing power

5.3.3 Procedure for RSA algorithm in Arduino

- Connect the Arduino to the computer via Arduino cable.
- Set the com number where the Arduino is connected.
- Upload the algorithm
- Compile the algorithm, enter the key, enter message and run it.
- Measure the processing power

5.3.4 Procedure for RSA algorithm in Raspberry pi

- Setup the Raspberry pi using relevant components such HDMI, power supply adapter and memory card.
- Upload the algorithm and compile it
- Run the algorithm, enter the encryption key and the message
- Measure the processing power

6 Results

The results that were obtained during the experiments test are as follows:

6.1 Functionality of the security algorithm

For AES security algorithm, the code was compiled and the user had to enter the size of the key that is desired for protection. Thereafter the plain text was also entered and the algorithm was executed. Figure 6.1 shows the results of the encryption process of the AES security algorithm.

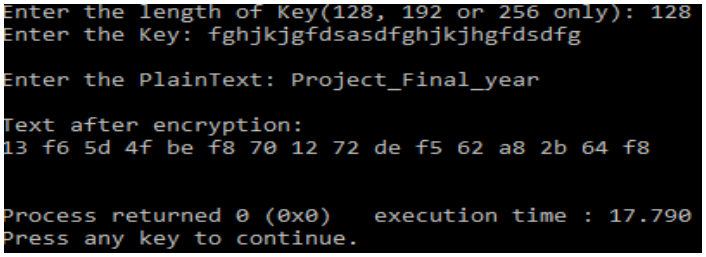

Figure 6.1: Output of AES security algorithm

For RSA security algorithm, the algorithm was uploaded and compiled in the software development. The user was required to enter two large prime numbers as the key. Thereafter the user should enter message to be encrypted. The algorithm automatically executes if the every entered information is correct. Figure 6.2 shows the results of RSA algorithm

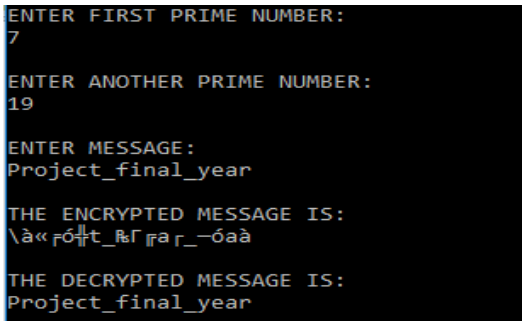

Figure 6.2: Output of RSA algorithm

6.2 Speed of encryption/decryption

This experiment was performed due fact that the security algorithms use different mathematical formula and theorems to encrypt and decrypt data. The relation of the objective of this experiment to the main objective of the project is that for IoT applications it is required that when a command is sent to IoT devices, the response of the IoT activity should be immediate, thus the response of the IoT application also depends on how long the encryption/decryption process takes. The results are in Table 1.

Table 1: Comparison in execution between RSA and AES algorithm

Data sizes (bytes)	**RSA security algorithm**	**AES security algorithm**
26,56	4	3
38,008	7	5
45,84	9	7
158,4	21	16
164,8	24	19
191,5	28	24
Average time	15,5	12,33
Byte/sec	6,72	8,44

6.3 CPU usage of the security algorithm

This experiment was done due to the fact that security algorithms consume lot of processing power, thus knowing the CPU usage for different security algorithms will provide knowledge to which optimum security algorithm should be used for securing IoT applications. The experiments were carried out on two micro-controllers. Each micro-controller has its own setup and software development. The results of the CPU usage are shown in Table 2 and Figure 6.3.

Table 2: CPU usage of the security algorithms

Security algorithm	**Arduino Uno**	**Raspberry pi**
AES	10,1 %	4.1%
RSA	19%	10%

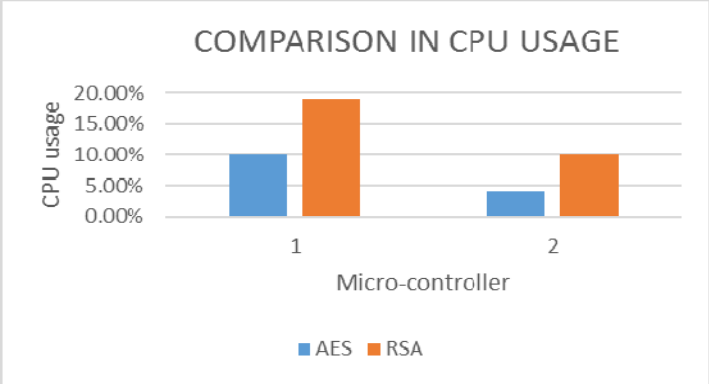

Figure 6.3: CPU usage of the security algorithms

7 Business Benefits

The project was aimed to provide security to newly existing IoT technology. The achievement of the project assures safety and confidentiality to people who use IoT applications, the achievement of the project however will also attract many people in using the technology, and this means that companies that provide service for IoT application will gain more customers and make more profit.

8 Conclusions

Initially, developing security algorithms was aimed to provide security for IoT applications. The security algorithms were planned to be able to encrypt/decrypt, authenticate and provide confidentiality. However authentication was not possible, this is due to the fact that authentication algorithms were tempering with the structure of the security algorithms and this led to leaving them out as they can affect the success of the project. Another shortcoming was the inability of the security algorithm to encrypt/decrypt different type's data formats such as sound, picture and video. This was due to the fact that the mathematical functions used security algorithms were not catered to covert such formats. To be able to overcome such shortcomings one should conduct proper research to avoid false results.

The ability of the security algorithms to generate key, encrypt and decrypt was possible. However looking at the obtained results AES security algorithm seem to be performing better than RSA security algorithm. It can be said that it was expected that the AES security algorithm would perform better than RSA a security algorithm. This is due to mathematical formulas, theorems and computation.

The future work of the project would be focusing on improving the performance of design 1 i.e. AES security algorithm. This is because it performs better than RSA algorithm as it has good security with low key size. The second focus should be finding ways of managing the key distribution. This is because AES security algorithm uses symmetric key i.e. the participants use identical private keys.

References

[1] D. Evans, "The Internet of Things," How the next Evolution of the internet is changing everything, p. 11, April 2011.

[2] J. B. W. a. M. P. Teng Xu, "Security of IoT Systems," no. Design Challenges and Opportunities, p. 7, 2014.

[3] N. B. a. M. Z. Andrea Zanella, "Internet of Things for smart cities," p. 11, February 2014.

[4] W. Diffie, Multi-user cryptographic techniques, p. 50, 08 June 1976.

[5] N. Hajdarbegovic, "Internet of Things Security," p. 15, March 2015.

[6] K. S. Muruganandam, "Data Encrytion and Decryption using Triple DES and Perfomance analysis," vol. 2, no. 11, p. 8, November 2014.

[7] A. Dhir, "Data Encryption using DES/Triple DES Functionality in FPGAs," vol. 1, p. 14, March 2000.

[8] V. R. Joan Daemen, "AES proposal," p. 47, April 2003.

[9] A. Berent, "Advanced Encryption Standard example," vol. 1.7, p. 16, 2010.

[10] N. Y. Goshwe, "Data Encryption and Decryption using RSA algorithm in network enviroment," vol. 13, p. 5, July 2013.

[11] B. Kaliski, "The methamatics of the RSA Public-Key Cryptosystem," p. 9, 2008.

[12] F. Information, "Specification for Advanced Encryption Standard," 26 November 2001, p. 51.

[13] T. Davis, "RSA encryption," p. 6, 10 October 2003.

[14] E. Mianov, "The RSA algorithm," p. 11, 3 June 2009.