International Journal of Computer Networks and Applications (IJCNA)
DOI: 10.22247/ijcna/2022/211632 Volume 9, Issue 1, January – February (2022)

RESEARCH ARTICLE

Design of Maintaining Data Security on IoT Data Transferred Through IoT Gateway System to Cloud Storage

S. Alexander Suresh
Department of Computer Science, Bishop Heber College (Affiliated to Bharathidasan University), Trichy, Tamil Nadu, India
salexandersuresh@gmail.com

R. Jemima Priyadarsini
Department of Computer Science, Bishop Heber College (Affiliated to Bharathidasan University), Trichy, Tamil Nadu, India
jemititus@gmail.com

Received: 24 January 2022 / Revised: 17 February 2022 / Accepted: 24 February 2022 / Published: 28 February 2022

Abstract – IoT is a smart-device environment; it acts smarter and gives higher quality to enhance the user experience. IoT is an open network environment. However, IoT's security challenges are more vulnerable because the devices are openly accessible to all in the network. IoT communication can be formed in different ways, where the device generates the data is communicated to other devices or gateway or cloud. Therefore, security challenges are in everywhere IoT environment. IoT Communication considered in this paper is gateway communication. Data generated from the IoT environment are directed to the cloud over a gateway system. Hence, the data essentials to be protected from the IoT devices to the gateway system and the gateway system to the cloud storage. The cloud is a public environment that makes security is a challenge to keep data secure. Therefore, securing IoT data to the cloud environment is the most significant focus. This paper focuses on securing the data from the gateway system to the cloud storage. The paper proposes an enhanced modern symmetrical encryption to secure data to ensure data security in the travel and storage of IoT data to the cloud. The proposed Enhanced Modern Symmetric Data Encryption (EMSDE) is a block cipher encryption technique. It encrypts data by 64-bit block. Proposed encryption is tested according to time and level of security. Thingspeak stores the IoT data sent from the IoT gateway system for implementation and testing. Based on the testing, the proposed EMSDE produces better results, and it is tailored to secure data stored in the cloud from IoT devices.

Index Terms – IoT-Cloud, IoT Gateway, Data Security, Cryptography, Symmetric Key, Encryption, Key Generation.

1. INTRODUCTION

The Internet of Things (IoT) grows rapidly in social, technical and economic significance [1]. Internet of Things can change us. Sometimes it seems unrealistic and vague, but the impact of IoT in the previous few years has been remarkable. IoT also called a group of internet objects interconnected for internet services shall have the capability to change everything. IoT is a distributed physical networking device intelligently detecting information gathered from environmental settings[2].

Furthermore, these devices can communicate information and messages to one another. The notions of the Internet of Things are like thinking outside the box. For example, an IoT chair can be adjusted depending on the body size of the person occupying it [3]. Likewise, an alarm clock may be altered, considering the person's health problems of the recent day and the activity planned for the following day. These calculations indicate that it may automatically increase or decrease sleep hours [4]. To change people's way of life, increase productivity and reduce the stress of life to relieve the environment. The IoT is a new generation of networks and a new moderate form of the Internet for highlighting the ultimate wisdom. IoT devices can collect a great deal of data, analyze it, make intelligent decisions, and redistribute information to devices to act intelligently [5].

In general, Internet of Things-based detection apps generates a significant amount of data that needs to be processed and stored. Because of IoT devices' limited data processing and storage capabilities, data processing and storage capabilities are largely transferred to the cloud [6]. The IoT is connected to the cloud for data storage, as indicated by the IoT-Cloud. Cloud Computing features limitless virtual storage and process data received from sensors. There are two convergences for IoT and Cloud i)

RESEARCH ARTICLE

Cloud-based IoT brings IoT capabilities to the cloud and ii) IoT-based cloud that brings cloud capabilities to IoT [7]. The cloud maintains data received from IoT devices, and it limited when the network has set the boundary and limited access, but while it comes to IoT, it becomes broader and more vulnerable [9]. The IoT devices have brought a new stack of security issues to be lined up. The hackers are provided more leverages because open access to monitor the data feeding, setting change, remote access, and authorization changes are easily accessible [10]. Data Security and network security play a vital role in the IoT during data transfer. IoT architecture comprises three layers: perception, application, and network [11]. Safety organization in the IoT is significant to consumers and designers as it permits them to sort improved conclusions in the selection, development and maintenance of IoT devices [12].

must be protected against illegal usage. Safeguarding data security in the cloud is challenging [8]. The challenges are

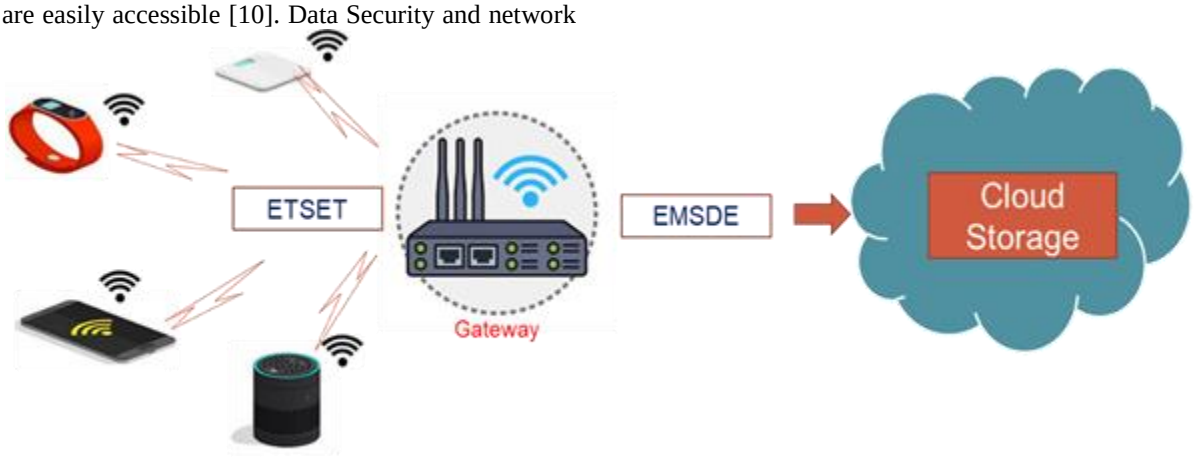

Figure 1 Proposed Abstract Research Framework

The entire research aims to secure IoT data communication through a gateway system. Figure 1 shows the proposed research framework. According to the framework, IoT data communication must be secured in two levels: securing the data from IoT device to gateway system and gateway system to cloud storage. The data from the IoT objects/Sensors is sent to the gateway system, secured by an already proposed technique called ETSET [13]. ETSET is an enhanced technique from TEA (Tiny Encryption Algorithm). It is designed according to the consideration of running in low-computation devices. It is running in the IoT devices to encrypt the data. The ETSET is used to secure the data between IoT devices to the gateway system. It is a lightweight encryption technique. The encrypted data by the ETSET is received at the gateway. But, sending this encrypted data to the cloud is vulnerable. Because ETSET is a lightweight technique designed to execute in IoT devices. Cloud is a more challenging place to secure the data. This paper aims to secure IoT data communication from the gateway system to cloud storage to provide more security on IoT data. The encrypted data in the gateway system is again encrypted using the proposed EMSDE. The EMSDE runs in the gateway system and encrypts already encrypted data by the ETSET.

The rest of the article is structured as follows. Section 2 describes the cryptography techniques used in IoT Security, and Section 3 elaborates related work. Section 4 discusses the methodology used in the proposed work. In Section 5, a detailed explanation of the proposed work is given. Section 6 describes the implementation setup used in the research work. Section 7 presents the results and discussion. Finally, section 8 concludes the paper.

2. CRYPTOGRAPHY TECHNIQUES IN IOT-CLOUD SECURITY

Cryptography focuses on securing data on the computer or storage network. Typical cryptography includes transposition and substitution encryption. Modern cryptography is divided into Stream Cipher and Block Cipher [14]. Block-based encryptions encrypt a fixed data block at a time. The length of a piece of data corresponds to the size of a piece of data. The encrypted data block is typically the same length as the unencrypted data block containing block ciphers. Stream-based encryption usually encrypts a single byte or a bit at a time. This is because stream encryptions produce and use one keystream for encryption, not just one key. In general, a stream cipher is much quicker than block encryption. This is due to the basic mathematical formulas used with stream ciphers [15].

Cryptography is usually considered symmetrical and asymmetrical (commonly known as public-key cryptography) [16]. Both encryption and decryption processes have one pair of keys in asymmetrical encryption. Between both keys, one key is referred to as the public key, and the other is referred to as the private key. If a plaintext message is encrypted by one

RESEARCH ARTICLE

of the keys in the pair called the public key, then the encrypted message is decrypted only another key in the pair. However, asymmetrical key algorithms do not run as fast as functions [17]. Consequently, asymmetric key algorithms are not as widely used as symmetrical counterparties. The following two are the major asymmetrical techniques: RSA and Diffie-Hellman.

RSA: It was developed by Rivest Shamir Adelman and was established in 1978. It is the first asymmetrical algorithm extensively used for signature and data encryption. It can be used with a key size of 768 and 1024 bits. This algorithm relies on a three-phase of execution. The first phase is the key generator. Prime numbers are used to produce the key in RSA. The second phase of the process has to do with encryption. Since it is an asymmetric cipher, encryption is performed by the public key of the pair of keys. The third phase of the RSA is decryption by the key private key in the pair. The process symmetrical key algorithms. In addition, asymmetrical key ciphers are commonly more complex and more sophisticated

is complicated. As a result, it may only be used for small data [18].

Diffie-Hellman(DH): The first well-known asymmetric-key algorithm in cryptography. The main objective of this algorithm is to share the keys between two end-points. Although symmetrical key algorithms are fast and safe, key sharing is continually challenging. In DH, initially, find a path to acquire the secret/private key for every system in the network. Once it finds a path, it establishes a secured communication to get the keys. Then, the communication entities use this secure communication path to share the secret/ private key. The shared private key then acts as the symmetrical encryption between both systems.

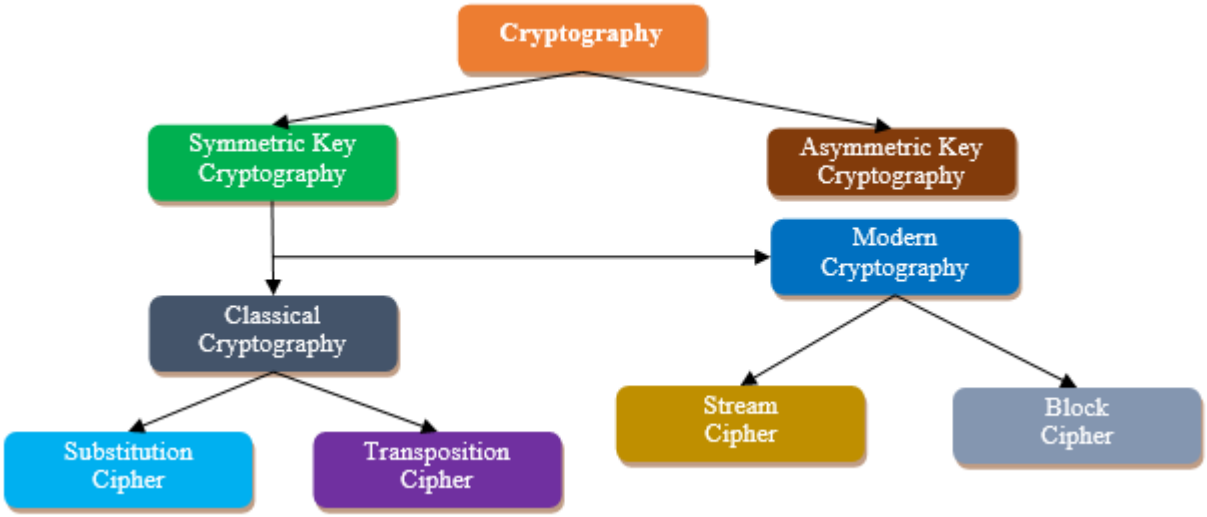

Figure 2 Cryptography Approach Categories

Symmetric key cryptography is categorised as conventional cryptography and modern cryptography. Figure 2 illustrates the cryptographical approaches used in reality data security applications.

Symmetrical key algorithms are also called secret/private key algorithms. Those approaches generally use a secret key, also called private which should be kept secure by the communication entities in the network. Symmetric-key ciphers are highly secure and fast. Further symmetric ciphers are more secured than asymmetrical ciphers in cryptography. It is practically unbreakable when the data are encrypted using symmetrical encryption. Symmetrical key encryption ciphers are also extremely speedy in computation. Because this feature is used frequently for conditions wherever there is a requirement to encrypt a large amount of data. Many different symmetrical natures of ciphers are available to secure the communication, and each has its own merits and demerits. Among all other symmetric ciphers, the following are the most familiar symmetric ciphers: DES, 3DES, AES, IDEA, RC4 and RC5 [19].

DES: It stands as Data Encryption Standard. It was invented in the year of 1976. It is one of the more commonly used symmetric encryption ciphers. The DES algorithms themselves are quite powerful. The shortcoming is that from the key size used in the DES. The standard DES cipher uses a 56-bit key for encryption and decryption. The pitfall of DES is the size of the key; it can

RESEARCH ARTICLE

make it easily breakable when using a computer to run all combinations of the bits in the key until it finds the correct one. Breaking the key was difficult when the DES was initially launched in security, but nowadays, using high power computers is done. This is the major reason the DES is not used much in security.

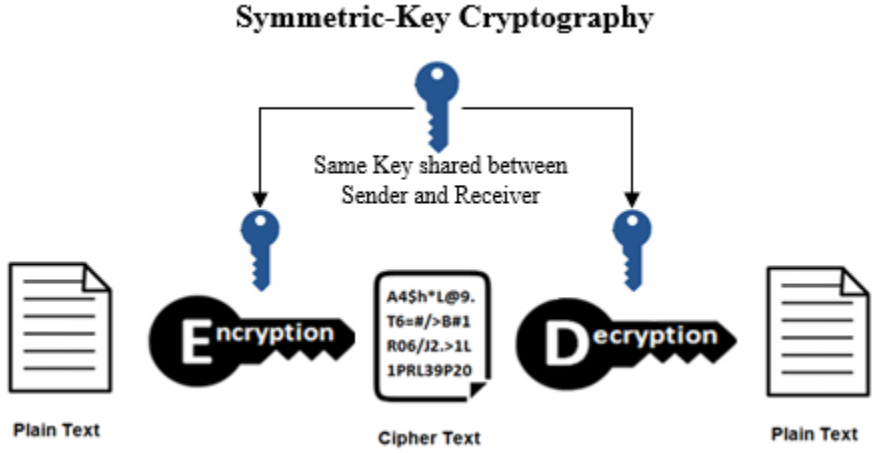

Figure 3 Symmetric Key Cryptography[20]

3DES: This is the next version of the DES cipher. This cipher is also called Triple DES. It gets this name by using the DES three times like Encryption-Decryption-Encryption. It provides better security than its predecessor, and it is the most used cipher in the early days instead of DES cipher. The 3DES cipher is more efficient than DES because it uses longer size keys. The safest execution of 3DES is to apply different keys for every iteration. When using a similar key for all the iterations of 3DES, then the size of the key is 56-bit. 3DES has three iterations when using the same key for two iterations and a different key for the third iteration; then, the key size is 112-bit. When using three different keys for three iterations, the encryption strength is 168-bit key.

AES: This is advanced cryptographic encryption. It stands as Advanced Encryption Stanard, and it is also often designated by the Rijndael. It's because AES is based on Rijndael's algorithm. It is established to replace the DES. It is more powerful and fast than the other encryption ciphers. The US govt selects AES after completing many reviews and evaluations. AES can be used in three different keys, and names are given according to AES 256, AES 192 and AES 128. The numeric value denotes the length of the key used in the specific version of the AES.

IDEA: This is the international data cipher algorithm. The original purpose of this algorithm was to substitute for the DES cipher. The size of the key used for IDEA is 128-bit. It is fast than DES, but it is not familiar as DES reached. IDEA is not used because of some reason. IDEA uses low length keys for encryption and decryption. It is faster than other algorithms, but it can't produce higher security than other ciphers.

RC4: RC4 stands Rivest Cipher. It is the fourth version of Rivest Cipher approaches. It uses the variable-sized key for encryption and decryption. The size of the key is in the range between 40-bit to 256-bit. The most generally used key size in RC4 is 128-bit. This cipher is not used as much even though it is simple and easy to apply. Because it is weak the cryptography execution and produces low security, it was slowly washed out in the security field.

RC5: it is similar to RC4 and the next release from RC4. It also uses variable-length keys. It can use up to a 2040-bit key. However, the most commonly used key length is 128-bit. It is also not used as much in the field of security. The symmetrical cryptosystem is more efficient for encrypting a large amount of data. The only burden in this approach is how the secret key is shared between the two end parties in the communication. The theory describes using a trusted third party to share the keys. Otherwise, the key can be manually distributed between communication partners in the network channels. Figure 3 depicts symmetrical key cryptography.

Symmetric key cryptography is used extensively to secure network data. This is the most common technique for securing IoT-cloud data. A cryptographic system whereby the source and destination of the data communication can share the unique secret key used for encrypting and decrypting the data [20]. Encryption is when encrypted characters on a fixed system replace raw text units. For example, data units may be single (most common) letters, pairs of letters, triple letters, combinations of the above [22].

A cryptographic method that applies a deterministic

RESEARCH ARTICLE

technique and a symmetric key to encrypt a data bit block is symmetric block-by-block encryption [23]. There are many symmetrical cryptosystems recommendations to use for IoT-Cloud data security. But running on the IoT environment requires an upgraded symmetrical key technique to secure IoT data. Therefore, the proposed research develops an appropriate symmetric key cryptography technique for IoT-cloud environments [24] [25].

3. RELATED WORK

IoT security is the most important preoccupation today. Wherever IoT intelligent devices have entered and made the environment more intelligent, this makes everything more intelligent; simultaneously, users wonder whether the data generated from IoT devices is secure or not. Data from IoT is outsourced to the cloud for storage, and it has been observed that storage in the cloud is not secure. It has reached the initial stage concerning the safety and confidentiality of user data [26]. Numerous research papers have been published with the solution for data security in the IoT environment, and few of them have attempted to solve security problems in IoT-Cloud. This section summarizes some of the associated solutions previously submitted by different researchers.

Jayapandian et al. [27] proposed a symmetric-key cipher and also an asymmetric-key cipher technique to ensure data security in the cloud. In this approach, symmetric-key cipher uses Data Encryption Standard to afford data protection using the single secret key. On the other hand, the asymmetric-key cipher uses the method pair of keys for encrypting and decrypting the data to maintain a secured cloud-based IoT data storage. However, the authors did not describe when and where both cryptographic systems are processed, and the specification of the symmetric and asymmetrical algorithm was not specified.

J D. Bokefodea et al. [28] delivered a that permits the organization to securely maintain the IoT device data in the cloud through various access control policies and cryptographic concepts. IoT devices are positioned to collect data in this architecture. A gateway node is an intermediary controller between IoT devices and the cloud server to maintain uninterrupted connectivity. In this architecture, the roles are defined for different users in the organization based on their responsibility. The admin controls the accessibility of the cloud server according to the necessity of the users. The admin appoints role managers to manage the users in the organization. The data collected from the devices is scrambled by the admin and saved in the cloud server for the specific role of users. Henceforth, only authorized users with suitable roles can retrieve the data by decrypting it and using it. It is a role-based cipher method.

A.Vithya Vijayalakshmi et al. [29] developed a multi-level encryption technique to improve IoT data security. In this approach, data detected from IoT devices is encrypted in the gateway using Merkle-Hellman and Elliptic Curve Cryptography (ECC) encryption for data security. The proposed technique aims to ensure data security detected by IoT devices. Data from IoT devices is transmitted to the gateway using protocols such as CoAP and HTTP over the Internet. Once the gateway receives the data, it is prepared to be passed to the storage server in the cloud. Data is encrypted using multi-level encryption techniques before data is transmitted to cloud storage. Data encryption involves two steps. As a first step, the Merkle-hellman cryptosystem is used to encrypt the data. In the second step, the encrypted text generated by Merkle-hellman is input into ECC. The resulting encryption is then sent to the cloud server. This approach ensures data security and increases computing time.

Canteaut, A. et al. [30] aimed to guarantee data confidentiality at end-to-end within a smart IoT network. Confidentiality is guaranteed when a low-end sensor node collects data until it is used stored in the cloud server. The data in the cloud server can be used for data analytics and applied to an artificial intelligence system. The data needs to be secured when it is in the cloud server. Therefore, it gives importance to data security in the cloud by using the end to end encryption in the cloud to ensure data confidentiality. Implementing a lightweight and low power encryption method on IoT devices is necessary to secure the data. Trivium lightweight IV stream encryption is used to ensure the safety of IoT data.

K. Saini et al. [31] introduced a novel cloud security method that suggested the hybrid approach to use symmetric and asymmetric crypto-systems. The authors focused on the drawbacks of functions, size of memory and computation time of cryptographic techniques such as AES, RSA, and MD5 to improve the strength of the novel method. In this method, a pair of keys is used where the public key is used to encrypt, and the private key is used to decrypt the data. The critical characteristic of this approach was that the server could not know the message and, therefore, the plain text was never stolen. However, symmetric and asymmetrical encryption sequentially creates more time complexity for data uploaded to the cloud.

N. L. Kodumru et al. [32] used all AES, RSA, One Time Pad and other cryptographic algorithms to upload data to the cloud computing environment. This approach focused on comparing the three models described above. The security of the three models has been analyzed. It found that RSA and OTP were sufficient to store data in cloud storage. According to the analysis, the author stated that the time and space complexity of the RSA and OTP algorithms was less than that of other techniques. But generally speaking, the RSA presents more space and time complexity than other cryptographic techniques.

RESEARCH ARTICLE

Anuj Kumar et al. [33] proposed a cryptographic system to ensure greater data protection in cloud storage using a hybrid system. The proposed hybrid method uses the RSA asymmetric and DES symmetric in hybrid mode to ensure data computation time to encrypt a small quantity of data. The DES also takes more computation time to encrypt the data. Furthermore, both algorithms are more convoluted and cannot be executed on IoT devices. Therefore, this hybrid approach does not support the data security of IoT devices.

According to secured data transmission, Ding Li et al. [34] has researched and developed heterogeneous network resource management algorithms. Haiyun Ma et al. [35] proposed a new approach for encrypting personal information in the IoT in a cloud-based computing environment. Under security. While this approach can ensure data security, it is not an effective integration of RSA with DES. Since RSA is an asymmetrical cryptographic system, it takes a longer

IoT, confidentiality information can be divided into several sub-spaces based on properties and acquisition time. According to the stream cypher, the paper developed an information gathering encryption system model. Fatemeh Rezaeibagha et al. [36] proposed an effective distributed homomorphic cryptographic system designed for multi-user cloud-based IoT environments. For clarity, the schema requires a semi-reliable server that facilitates multiplicative Encryption calculation. This demonstrates that this scheme was semantically secure against potential opponents. Table 1 shows the comparison of the literature review.

Paper	Cryptosystem	Security Service	Level of Security Layer	Cipher	Issues
Jayapandian et al. [27]	Symmetrical	Confidentiality	Network Layer	Block	Key Management Issue
J D. Bokefodea et al. [28]	Asymmetrical	Authentication	Application Layer	Block	Extra Time
A.Vithya Vijayalakshmi et al. [29]	Asymmetrical	Confidentiality & Authentication	Application Layer & Network Layer	Block	Sequential execution of Algorithms takes more time
Canteaut, A. et al. [30]	Symmetrical	Confidentiality	Network Layer	Block	Additional overhead on computation
K. Saini et al. [31]	Symmetrical & Asymmetrical	Confidentiality & Authentication	Network Layer	Block	Extra Time
N. L. Kodumru et al. [32]	Symmetrical & Asymmetrical	Confidentiality	Network Layer	Block	Extra Time
Anuj Kumar et al. [33]	Symmetrical & Asymmetrical	Confidentiality	Network Layer	Block	Extra Time
Ding Li et al. [34]	Symmetrical	Confidentiality	Network Layer	Block	Large data blocks
Haiyun Ma et al. [35]	Symmetrical	Confidentiality & Integrity	Network Layer	Stream	Possible for Network Attack
Fatemeh Rezaeibagha et al. [36]	Symmetrical	Confidentiality	Network Layer	Block	Communication Error Occurs

Table 1 Comparison of Literature Review

4. METHODOLOGY OF EMSDE

Many researchers have tried to provide solutions to the security issues in the smart IoT environment. However, it is still not enough to resolve the security challenges in the entire IoT-cloud environment. The proposed EMSDE specifically ensures the security data between the gateway system to cloud storage. Figure 4 exemplifies the block diagram of the EMSDE proposed in this paper. The proposed EMSDE is a symmetric encryption technique. The key determination of

RESEARCH ARTICLE

this research is to secure data in the cloud environment. This paper presents an Enhanced Symmetric Data Encryption Technique to ensure the security of the IoT data. The proposed EMSDE is a block cipher encryption technique. It takes 64 bits of plain text as input and produces 64-bit ciphertext as output. EMSDE has eight rounds of routine to encrypt the data.

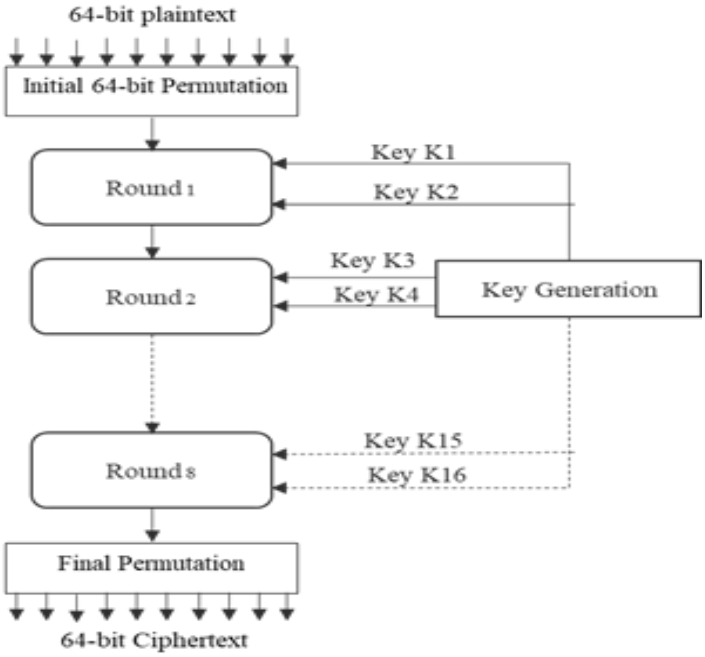

Figure 4 Block Diagram of EMSDE

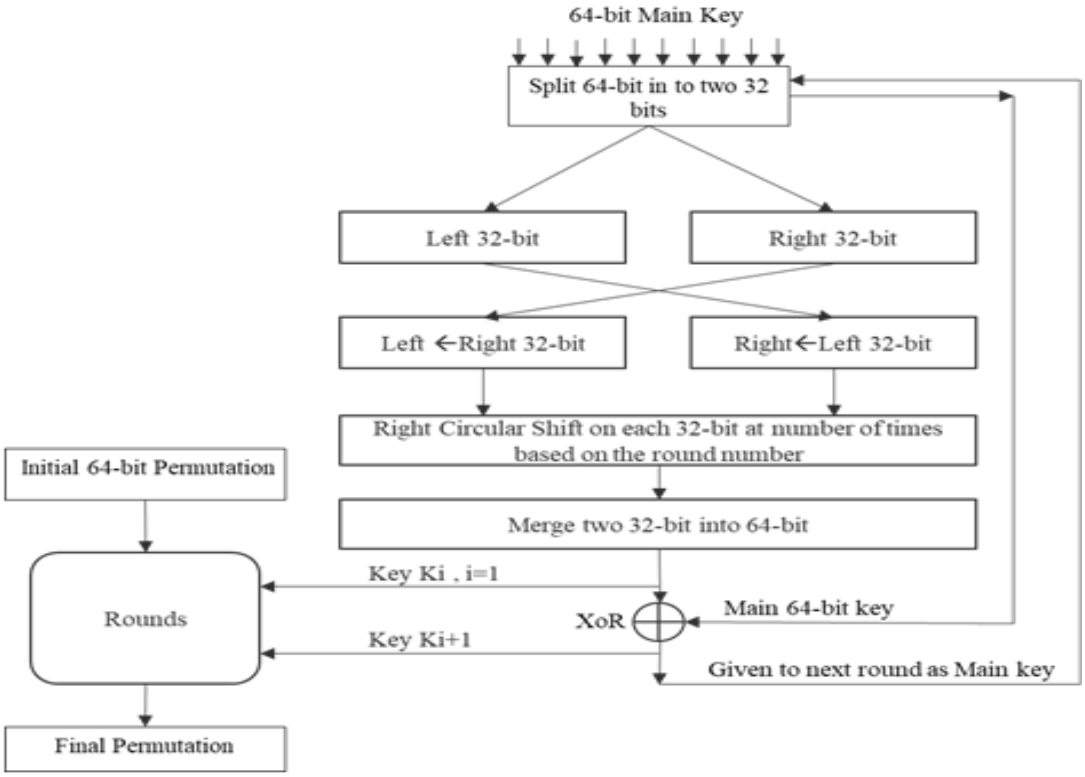

Figure 5 Process Flow of Encryption and Key Generation of EMSDE

RESEARCH ARTICLE

Each round takes two keys to encrypt the data. The size of the primary key is 64 bits. The main 64 bits key is processed and generates 16 subkeys, 2 keys per round basis. The proposed EMSDE is running in the gateway system of the IoT network. Gateway system connected with all the IoT devices in the network.

Data acquired by the smart IoT devices are encrypted in the device itself using a lightweight encryption technique already proposed in [13] named ETSET. Then, the encrypted information from the devices is forwarded to the gateway system. Gateway receives encrypted data from IoT devices. Gateway could not send this encrypted data to the cloud because a lightweight encryption technique encrypts it. The encryption is carried in the device itself. Hence, it secures the data when transmitted from the IoT device to the gateway. But, it is not ensured that lightweight encryption can also provide security in the cloud environment. Because the cloud is an open public environment, there are possibilities to use superpower computers to hack the data. Hence, the data is further involved in a conversion that ensures data protection when stored in the cloud.

Consequentially, EMSDE running in the gateway system encrypts the already encrypted data by the ETSET before data is sent to the cloud. Figure 5 shows the Encryption and key generation of EMSDE.

5. EXPLANATION OF EMSDE

EMSDE encrypts the data in the gateway system, ready to store in the cloud storage. EMSDE is a symmetric encryption technique. Figure 5 describes the step by step process flow of the proposed EMSDE. The following steps elucidate the proposed EMSDE Encryption and key generation execution.

5.1. EMSDE Encryption

The proposed EMSDE is a symmetric block cipher. It encrypts 64 bits of the data block at a time. Encryption running for 8 rounds. Initially, the 64 bits plaintext is given as input. Then, the 64 bits are taken into an 8*8 matrix of the table. Finally, a permutation of 64 bits is carried out according to Figure 6. In Figure 6, the left side table bits are permutated to the right side table of bits. This permutation happens before the round function starts.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

64-bit Plantext

64	49	41	33	25	17	9	57
58	55	42	34	26	18	50	2
59	51	46	35	27	43	11	3
60	52	44	37	36	20	12	4
61	53	45	29	28	21	13	5
62	54	22	38	30	19	14	6
63	15	47	39	31	23	10	7
8	56	48	40	32	24	16	1

Intial 64-bit Permutation

Figure 6 Initial Permutation

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

64-bit Plantext

60	44	28	12	32	16	64	48
47	63	15	31	11	27	43	59
58	42	26	10	30	14	62	46
45	61	13	29	9	25	41	57
52	36	20	4	24	8	56	40
39	55	7	23	3	19	35	51
50	34	18	2	22	6	54	38
37	53	5	21	1	17	33	49

Row Shuffling Permutation

Figure 7 Row Shuffling Permutation

RESEARCH ARTICLE

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

64-bit Plantext

7	29	39	61	60	34	28	2
15	21	47	53	52	42	20	10
23	13	55	45	44	50	12	18
31	5	63	37	36	58	4	26
8	30	40	62	59	33	27	1
16	22	48	54	51	41	19	9
24	14	56	46	43	49	11	17
32	6	64	38	35	57	3	25

Final Permutation

Figure 8 Final Permutation

The result of permutation is given as input to the first round. Subsequently, the round result is given as input to the next round. In rounds, initially, 64 bits plaintext is XORed with key Ki, where i←1. After XoR, 64 bits are transposed according to the row shuffling permutation table shown in Figure 7. In Figure 7, the left side table bits are permutated to the right side table of bits.

Next, the 64-bit is split into two halves, Left 32-bit and Right 32-bit. Count the 1s in each 32-bit block. The left 32-bit is rotated clockwise the number of times up to the counting value of 1 compared to the left 32-bit. Right, 32-bit is rotated clockwise to the number of times to the counting value of 1 on the right 32-bit. The two 32-bit are merged and XORed with Key Ki+1. A round is completed, the result of a round 64 bits is given as input to the next round. Rounds are continued. After completion of the 8th round, it produces a 64 bits ciphertext. Finally, the 64-bit final permutation is carried out based on the tables shown in Figure 8. The key generator generates two subkeys from the primary key to the Encryption and Decryption of the proposed EMSDE.

5.2. Steps involved in the EMSDE Encryption Procedure

Step 1: IoT device's data are considered as input

Step 2: The plaintext is taken as binaries

Step 3: Find the total size of the binaries

Step 4: The proposed EMSDE consider a 64-bit block of input. Therefore, the total binaries are divided into 64 bits blocks.

Step 5: The 64bits block is permutated. The 64 bits are formed as an 8_X8 matrix. Then, the bits are permutated according to the initial permutation table.

Step 6: Round 1 started with the initial permutation of i64 bits. In the round, initially, the 64 bits are XoR with the Key K1.

Step 7: The result of XoR is applied by row shuffling permutation according to the row shuffling permutation table.

Step 8: The 64 bits are split into equal halves of 32-bit.

Step 9: Count the number of 1's in every 32 bits and rotate every 32 bits up to the number of 1's in every 32 bits. On the left rotation side, 32 bits are rotated left, and right side, 32 bits are rotated right.

Step 10: Both 32 bits are merged into 64 bits.

Step 11: The 64 bits are XORed with Key K2, and the result of this XoR is given as input to the next round.

Step 12: The round steps from Step 6 to Step 11 is repeated for eight rounds.

Step 13: After round 8, a 64 bits pre-cipher text is produced.

Step 14: The 64-bit is applied to the final permutation according to the final permutation table to produce the 64-bit ciphertext.

Step 15: The 64-bit ciphertext is converted to decimal and converted to character code to produce the ciphertext in character code.

The EMSDE is invoked for encrypting the data before sending the cloud. According to the pseudo-code given in this section, it is implemented using an Arduino microcontroller. The EMSDE encryption process is given in Pseudocode 1.

Sub EMSDE(plaintext)

1. Start
2. pt← 64 bits plaintext
3. inp←pt.tostring()
4. ptb[]← {64 49 4133 25 17 9 57
   58 55 4234 26 1850 2

RESEARCH ARTICLE

```
                59 51 4635 27 4311 3
                60 52 4437 36 2012 4
                61 53 4529 28 2113 5
                62 54 2238 30 1914 6
                63 15 4739 31 2310 7
                8 56 48 40 32 24 16 1}
        //Initial Permutation
5.      for i←1 to ptb.length
                pt[i]= inp.charAt(ptb[i]-1);
6.      next i
7.      r←1
        //Executes for 8 rounds
8.      while (r<=8)
9.              xpt←pt ⊕Km, m←1,2,3…16
10.             M←1
                //matrix 8x8
11.             for i← 1 to 8
                for j← 1 to 8
                        ptm[i][j]←xpt[m]
                        m←m+1
                next j
                next i
12.              inp←ptm.tostring()
13.             ptb[]← {60 44 2812 32 1664 48
                        47 63 1531 11 2743 59
                        58 42 2610 30 1462 46
                        45 61 1329 9 25 41 57
                        52 36 204 24 8 56 40
                        39 55 7 23 3 19 35 51
                        50 34 182 22 6 54 38
                        37 53 5 21 1 17 33 49}
                //Row shuffling permutation
14.             for i←1 to ptb.length
                        ptc[i]= inp.charAt(ptb[i]-1);
15.             next i
                //Split in to two half
16.             spt←ptc.to string()
17.             lpt←split(spt,0,31)
18.             rpt←split(spt,32,63)
19.             lpt and rpt is considered as a matrix
                //Count 1s in each block
20.             for i←1 to lpt.length
                        if(lpt[i]%2!=0)
                        lpt1++
                next i
21.             for i←1 to rpt.length
                        if(rpt[i]%2!=0)
                        rpt1++
                next i
22.             slpt1←split(lpt,0,lpt1)
23.             slpt2←split(lpt,lpt1+1,lpt.length)
24.             mlpt←merge(slpt1,slpt2)
25.             srpt1←split(rpt,0,rpt1)
26.             srpt2←split(rpt,rpt1+1,rpt.length)
27.             mrpt←merge(srpt1,srpt2)
28.             mpt←merge (mlpt,mrpt)
29.             xrpt←mpt ⊕Km+1
30.             pt←xrpt
31.             r++
32.     end while
33.     inp←xrpt.tostring()
34.     ptb[]← {7 29 39  61 60 34 28 2
                15 21 4753 52 4220 10
                23 13 5545 44 5012 18
                31 5 63 37 36 58 4 26
                8 30 40 62 59 33 27 1
                16 22 48 54 51 41 19 9
                24 14 56 46 43 49 1117
                32 6 64 38 35 57 3 25}
        //Final Permutation
```

RESEARCH ARTICLE

35. for i←1 to ptb.length

pt[i]← inp.charAt(ptb[i]-1);

36. next i
37. ciphertext←pt

end

Pseudocode 1 EMSDE Encryption Process

The proposed EMSDE encrypts data using a key. The EMSDE uses 16 keys to encrypt and decrypt data. The following section describes the key generation in the EMSDE.

5.3. Key Generation

The EMSDE is given key input is 64 bits. It is the main key K, and it is generated for 64 bits of EMSDE encryption. From this 64 bit primary key, 16 subkeys are generated based on two keys per round. The key generation parallelly executes to generate the subkeys for each round. In the subkey generation, initially, key K is divided into equal halves of 32 bits. Then, the left and right 32 bits are rotated left at the number of times based on the round number. It is rotated one time for the first round, the second round, rotated two times, and so on. Finally, the two 32 bits are merged and generate 64 bits key Ki, i←1,2,3…16. Ki is the first key for the first round of EMSDE encryption. The Key Ki is XORed with Key K and produces another Key Ki+1, which is used as the second key for the same round in EMSDE encryption. Two keys are generated for each round. Sixteen subkeys are generated for a 64 bits block of encryption. The detailed procedure of key generation is given below.

5.4. Steps involved in the EMSDE Key Generation

Step 1: Initial key Ki is 64-bit taken as input.

Step 2: The Ki 64-bit is split into equal two halves, left 32-bit and right 32-bit

Step 3: Perform 32-bit swap

Step 4: The 32 bits are rotated circularly right-side based on the round number. The bits are rotated the number of times equal to the round number.

Step 5: Each rotated 32 bits are merged as 64 bits and considered key Ki+1, i←0.

Step 6: Find the XoR for Ki and Ki+1 and produce Ki+2

Step 7: The Key Ki+2 is considered the second key for the same round.

Step 8: The Key Ki+2 is input to produce the Ki+3 for the upcoming rounds.

Step 9: Steps 2 to 9 are rounded according to the proposed EMSDE encryption and decryption.

The proposed EMSDE is a symmetric encryption technique. It uses 16 subkeys for eight rounds. Each round uses two subkeys to substitute the new bits using the XoR operation. The ciphertext produced using the proposed EMSDE is decrypted using the same 16 subkeys used in the encryption. The EMSDE Key generation process is given in Pseudocode 2.

Sub EMSDEkey(plaintext)

1. Start
2. mky← 64 bits key
3. round number

//Split into two half

4. do
5. sky←mky. to string()
6. lky←split(sky,0,31)
7. rky←split(sky,32,63)
8. swap(lky,rky)

//right rotation each blocks

9. lky1←split(lky,0,r)
10. lky2←split(lky,r+1,lky.length)
11. mlky←merge(lky1,lky2)
12. rky1←split(rky,0,r)
13. rky2←split(rky,r+1,rky.length)
14. mrky←merge(rky1,rky2)
15. kym←merge(mlky,mrky)
16. Key Km←kym // key k1 for a round

//XoR for Next key

17. Km+1←mky⊕kym // key k2 for the same round
18. mky←kym
19. while(r<=8)

end

Pseudocode 1 EMSDE Key Generation Process

5.5. EMSDE Decryption

Decryption is an inverse procedure of encryption procedures. The decryption process takes ciphertext as the input to convert it into plain text. The identical keys are applied for encrypting and decrypting the data. The keys are used in reverse order. For example, the Keys K1 to K16 is used in encryption, but in Decryption, K16 to K1 is used from round1 to round 8.

RESEARCH ARTICLE

6. IOT-CLOUD ENVIRONMENT EXPERIMENTAL SETUP

The proposed EMSDE is a part of our entire research work. An IoT-cloud environment is set up for the experiment of the proposed entire research. To create the IoT environment, sensors, Arduino Board, Wifi controller etc., are used. According to the proposed research, data from the sensors are composed, encrypted and forwarded to the cloud. The list of components used in the experimental setup is given in Table 2.

S. No	Name	Quantity	Component
1.	U1	1	Arduino Uno R3
2.	U2	1	LM35 [TMP36]
3.	U3	1	Wifi Module [ESP8266]
4.	U4	1	AS [HC-SR-04]
5.	R1 R2	2	1 K Resistor

Table 2 Components Used in the IoT-Cloud Experiment

The Arduino Uno R3 is a microcontroller board is connected with two sensors called LM35 and AS. ESP8266 is used for sending data to the cloud. ThingSpeak is connected with a setup to receive the data in the place of the cloud. The proposed EMSDE encryption and key generation are coded in the Arduino microcontroller and managed by the IoT Gateway System. Figure 9 illustrates the configuration of the component and the connection of the experience environment.

The data received from the IoT devices are given as the input to EMSDE. The EMSDE efficiently encrypts the data given as input. The encrypted data after EMSDE encryption are forwarded to the ThingSpeak. From the output produced by the EMSDE, it is observed that the proposed EMSDE generates a different ciphertext for the similar inputs in the plaintext. But, the current encryption approaches are generated the identical ciphertext for the similar inputs in the plaintext. The ciphertext produced by EMSDE makes the cryptanalyst confused to generate the original text. Hence, it is tough for the cryptanalyst to find the plaintext by different security attacks. Therefore, the ciphertext becomes unintelligible data that adversarial cloud users could not read. The data can be retrieved only by authenticated users who have the right keys. Consequently, it is essential to hold on the keys sealed.

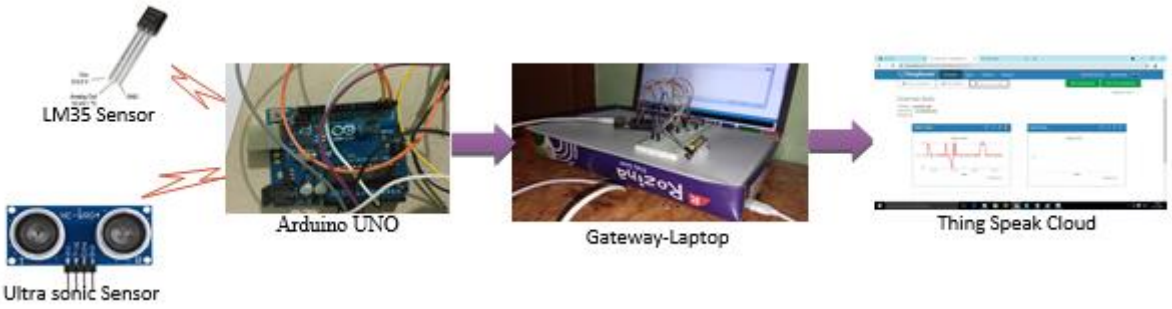

Figure 9 IoT-Cloud Experiment Setup

7. RESULTS AND COMPARISON

The performance and security level of the proposed EMSDE is measured alone with existing encryption techniques. The performance of the EMSDE is measured according to the time consumed for Encrypting and Decrypting the data. Dissimilar sizes of data accompany the experiment. First, time is calculated for various size data concerning Encryption, Decryption and evaluated for existing techniques. Then, the investigation is conducted at different intervals for the same data size, and the average is found and mapped with the other methods. Finally, the data are submitted to EMSDE for Encryption and Decryption in a cloud server.

Table 3 and figure 10 show the performance comparison of EMSDE concerning the time taken for encryption with existing techniques. The time consumed for encrypting the data by existing/present and proposed encryption techniques is measured for dissimilar data sizes. The results from the comparison depict that the proposed EMSDE has consumed a minimum time for encryption.

Size (KB)	DES	Blowfish	Proposed EMSDE
100	52	41	31
200	96	80	59
300	142	124	88
400	191	168	119
500	248	212	148
1 MB	492	429	302

Table 3 Performance Comparison Based on Encryption Time

RESEARCH ARTICLE

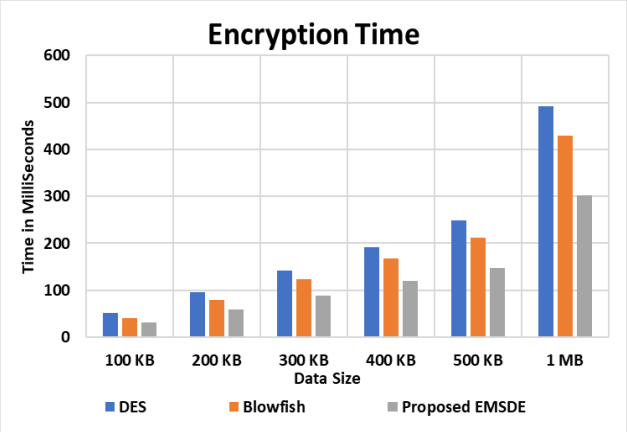

Figure 10 Comparison of Encryption Time

Table 4 and Figure 11 present the performance calculation and comparison of EMSDE and other techniques concerning decryption time. The time consumed by the existing/present and proposed decryption techniques is measured for dissimilar data sizes. The result shows from the table and graph that the proposed EMSDE technique has consumed a minimum time for decrypting data.

Size (KB)	DES	Blowfish	Proposed EMSDE
100	49	38	29
200	92	79	57
300	138	119	85
400	187	162	116
500	243	208	145
1 MB	485	415	294

Table 4 Performance Comparison Based on Decryption Time

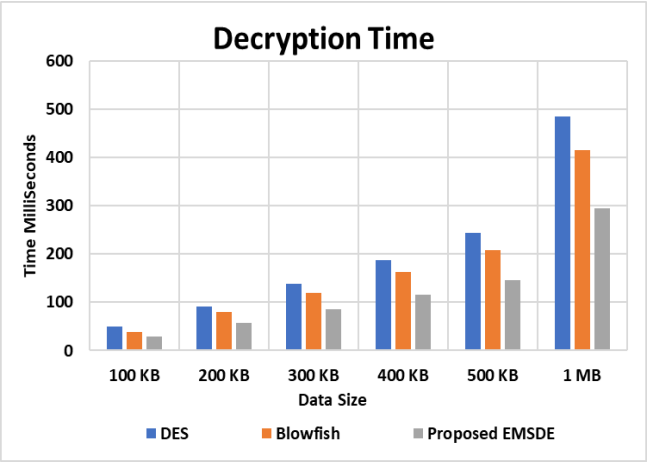

Figure 11 Comparison of Decryption Time

The security level proposed technique is measured by using the ABC Universal Hackman. It is a hacking tool used to hack the encrypted data in the server. It is deployed in the cloud server to hack the data encrypted by the EMSDE. The data encrypted in the gateway is uploaded to the cloud. This hacking tool tries to retrieve the original data from the encrypted data without knowing the key and plaintext. It can use different security attacks to hack the data. Both existing and proposed approaches are measured for security using this tool. Finally, based on the mathematical calculation, the hacking tool analyst the security level of the encryption techniques. The mathematical formula for computing the level of security for both encryption methods are as follows,

Let M be the ciphertext size stored in the Cloud storage; P mentions the plaintext hacked and retrieved from the encrypted text by the ABC Hackman tool.

To compute the security level of encryption techniques,

$$T \leftarrow M - P \quad \ldots(1)$$

Where T points to the number of texts not similar to the plain text.

Then, the level of security is calculated in percentage using the following formula,

$$SE \leftarrow T/M*100 \quad \ldots(2)$$

Where SE denotes the percentage of the security level of the proposed encryption technique.

S. No.	Encryption Techniques	Security Level (%)
1	DES	78
2	Blowfish	84
3	Proposed EMSDE	90

Table 5 Comparison of Security Levels of Existing and Proposed Encryption Techniques

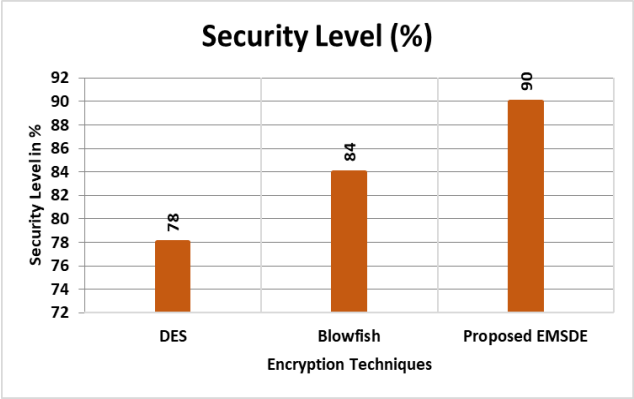

Figure 12 Security Levels for Different Encryption Techniques

RESEARCH ARTICLE

Table 5 and Figure 12 represent the evaluation of security levels for different encryption techniques. The result shows that EMSDE produces maximum security for the cloud data compared to the existing techniques. The security level of EMSDE is 90%, DES is 78%, and the Blowfish is 84%. EMSDE has a maximum level of security than other encryption techniques. Hence, EMSDE ensures the security of the IoT sensor data stored in the cloud server.

8. CONCLUSION

The EMSDE cryptographic technique is to protect data in a cloud-based IoT environment. The proposed EMSDE is a symmetrical nature of encryption that uses the single secret key for Encrypting and Decrypting the data. Encryption involves substitution and permutation of bits. The permutation is based on the permutation tables and the substitution of bits based on the XoR operations. EMSDE randomly produces the ciphertext for plaintext. Adversarial users are unable to hack the encrypted text produced by EMSDE. It makes confusion and diffusion to the cryptanalysis to hack the ciphertext in the cloud server. The experiment environment is set up for evaluating the research. The metrics used to measure effectiveness are time and security. Users' data are submitted to EMSDE with the keys. The data is encrypted before being stored in the cloud. The time for Encryption and Decryption is written down and compared to the existing technique. EMSDE takes less time for Encryption and Decryption than existing encryption techniques. The security level is determined by the ABC universal Hackman tool. The results indicate that the proposed EMSDE is more effective and ensures data security of IoT-Cloud data.

REFERENCES

[1] Panagiotis I. Radoglou Grammatikis, Panagiotis G. Sarigiannidis, Ioannis D. Moscholios, Securing the Internet of Things: Challenges, threats and solutions, Internet of Things, Elsevier, Vol. 5, 2019, pp. 41–70.

[2] Hossein Ahmadi, Goli Arji, Leila Shahmoradi, Reza Safdari, Mehrbakhsh Nilashi and Mojtaba Alizadeh, The application of Internet of things in health-care: a systematic literature review and classification, Universal Access in the Information Society, 2019, Vol. 18, pp. 837–869.

[3] Faizan Khursheeed, M Sami-Ud-din, Irshad Ahmed Sumra, Muhammad Safder, A Review of Security Mechanism in the Internet of Things, IEEE, 2020, pp. 1-19.

[4] Dewanjee, R., P. Verma, and R. Vjas. "Cryptography Techniques and Internet of Things." IEEE International Conference on Electronics and Communication Systems, 2016, pp. 1-4.

[5] Ali Hameed and Alauddin alomary, "Security Issues in IoT: A Survey", IEEE International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies, 2019, pp. 1-5.

[6] Subir Halder and Mauro Conti, CrypSH: A Novel IoT Data Protection Scheme Based on BGN Cryptosystem, IEEE Transactions on Cloud Computing, 2021, pp. 1-14.

[7] Preeti Mishra, Sachin Kumar, Umang Garg, Emmanuel S. Pilli and R.C. Joshi, Security Perspectives of Various IoT Cloud Platforms: A Review & Case Study, IEEE International Conference on Computing, Communication, and Intelligent Systems, 2021, pp. 727-731.

[8] Raja S., Manikandasaran S.S., Doss R., "Threat Modeling and IoT Attack Surfaces". In: Aurelia S., Paiva S. (eds) Immersive Technology in Smart Cities. EAI/Springer Innovations in Communication and Computing. Springer, Cham. 2022, pp. 229-258.

[9] Nickson M. Karie, Nor Masri Sahri and Paul Haskell-Dowland, "IoT Threat Detection Advances, Challenges and Future Directions", IEEE Workshop on Emerging Technologies for Security in IoT, 2020, pp. 22-29.

[10] Ali Hameed and Alauddin alomary, Security Issues in IoT: A Survey, IEEE International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies, 2019, pp. 1-5.

[11] C. Linda Hepsiba, Dr. R. Jemima Priyadarsini, Dr. S.Titus, A Comprehensive Study on Routing Attacks with Countermeasures in the Internet of Things, Solid State Technology, Volume: 63 Issue: 4, 2020, pp. 1-7.

[12] Jeethu Mathew, Dr. R. Jemima Priyadarsini, A Review on DoS Attacks in IoT, Solid State Technology Volume: 63 Issue: 4, 2020, pp. 1-10.

[13] S. Alexander Suresh SDB, Dr. R. Jemima Priyadarsini, ETSET: Enhanced Tiny Symmetric Encryption Techniques to Secure Data Transmission among IoT Devices, Turkish Journal of Computer and Mathematics Education, Vol.12 No.10, 2021, pp. 1094- 1099.

[14] Ding Li, Wang Zhongsheng, Wang Xiaodong, Wu Dong, Security information transmission algorithms for IoT based on cloud computing, Computer Communications, Elsevier, Vol. 155, 2020, pp.32-39.

[15] Shancang Li, Chapter 2 - Security Architecture in the Internet of Things, Editor(s): Shancang Li, Li Da Xu, Securing the Internet of Things, Syngress, 2017, pp. 27-48.

[16] Muhammad Sheraz Mehmood, Muhammad Rehman Shahid, Abid Jamil, Rehan Ashraf, Toqeer Mahmood, Aatif Mehmood, A Comprehensive Literature Review of Data Encryption Techniques in Cloud Computing and IoT Environment, IEEE, 2019, pp. 54-59.

[17] Roderick Hodgson, Solving the security challenges of IoT with public-key cryptography, Elsevier Network Security, Vol. 2019, Issue 1, 2019, pp. 17-19.

[18] Chandrasegar Thirumalai, Senthilkumar Mohan, Gautam Srivastava, An efficient public key secure scheme for cloud and IoT security, Elsevier Computer Communications, Volume 150, 2020, pp. 634-643.

[19] Shancang Li, Chapter 4 - IoT Node Authentication, Editor(s): Shancang Li, Li Da Xu, Securing the Internet of Things, Syngress, 2017, pp. 69-95.

[20] Symmetric vs Asymmetric Encryption – What are the differences? Accessed on 04 Nov 2021, https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences.

[21] Susha Surendran, Amira Nassef, Babak D. Beheshti, A Survey of Cryptographic Algorithms for IoT Devices, IEEE, 2018, pp. 1-8.

[22] Kejun Chen, Shuai Zhang, Zhikun Li, Yi Zhang, Qingxu Deng, Sandip Ray and Yier Jin, Internet-of-Things Security and Vulnerabilities: Taxonomy, Challenges and Practice, Springer Journal of Hardware and Systems Security, Vol. 2, 2018, pp. 97-110.

[23] Hossein Ahmadi, Goli Arji, Leila Shahmoradi, Reza Safdari, Mehrbakhsh Nilashi and Mojtaba Alizadeh, The application of Internet of things in health-care: a systematic literature review and classification, Springer Universal Access in the Information Society, 2018, pp. 1-33.

[24] Debabrata samanta , Ahmed h. Alahmadi, Karthikeyan m. P., Mohammad Zubair khan, Amit Banerjee, Gautam Kumar Ganapa, and Seeram Ramakrishna, Cipher Block Chaining Support Vector Machine for Secured Decentralized Cloud-Enabled Intelligent IoT Architecture, IEEE Access, Vol. 9, 2021, pp. 98013- 98025.

[25] Neha Kashyap, Ajay Rana, Vineet Kansal, Himdweep Walia, Improve Cloud-Based IoT Architecture Layer Security - A Literature Review, IEEE International Conference on Computing, Communication, and Intelligent Systems, 2021, pp. 772-777.

[26] Mohd. Tajammul, Rafat Parveen, Auto encryption algorithm for uploading data on cloud storage, International Journal of Information Technology, Springer, Vol. 12, Issue 3, 2020, pp. 831–837.

[27] N. Jayapandian, A. M. J. M. Z. Rahman, S. Radhikadevi and M. Koushikaa, "Enhanced cloud security framework to confirm data

RESEARCH ARTICLE

security on asymmetric and symmetric key encryption," 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), Coimbatore, 2016, pp. 1-4.doi: 10.1109/STARTUP.2016.7583904

[28] Jayant D. Bokefodea, Avdhut S. Bhiseb, Prajakta A. Satarkara and Dattatray G. Modanic, Developing A Secure Cloud Storage System for Storing IoT Data by Applying Role-Based Encryption, Twelfth International Multi-Conference on Information Processing-2016 Procedia Computer Science 89 (2016) 43 – 50

[29] A.Vithya Vijayalakshmi and Dr L. Arockiam, Enhancing The Security Of Iot Data Using Multilevel Encryption, International Journal of Advanced Research in Computer Science, Vol. 8, Issue 9, Nov–Dec 2017, pp. 841-845.

[30] Canteaut, A., Carpov, S., Fontaine, C., Jacques, Fournier, Lac, B., Naya-Plasencia, M., Sirdey, R., & Tria, A. (2017). End-to-end data security for IoT: from a cloud of encryptions to encryption in the cloud. In Proc. IEEE Conf. (Cesar), Nov. 2017, pp. 1–21.

[31] K. Saini, V. Agarwal, A. Varshney and A. Gupta, "E2EE For Data Security For Hybrid Cloud Services: A Novel Approach," 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida (UP), India, 2018, pp. 340-347. doi: 10.1109/ICACCCN.2018.8748782

[32] N. L. Kodumru and M. Supriya, "Secure Data Storage in Cloud Using Cryptographic Algorithms," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 2018, pp. 1-6.doi: 10.1109/ICCUBEA.2018.8697550.

[33] Anuj Kumar, Vinod Jain and Anupam Yadav, "A New Approach for Security in Cloud Data Storage for IoT Applications Using Hybrid Cryptography Technique", IEEE, 2020, pp.514-517.

[34] Zejun Ren, Xiangang Liu, Runguo Ye, Tao Zhang, Security and Privacy on Internet of Things, IEEE, 2017, pp. 140-144.

[35] Haiyun Ma and Zhonglin Zhang, A New Private Information Encryption Method in the Internet of Things under Cloud Computing Environment, Wireless Communications and Mobile Computing, 2020, pp. 1-9.

[36] Fatemeh Rezaeibagha, Yi Mu, Ke Huang, Lanxiang Chen, Leyou Zhang, Toward Secure Data Computation and Outsource for Multi-User Cloud-Based IoT, IEEE Transactions on Cloud Computing, 2021, pp. 1-12.

Authors

Mr. Alexander Suresh, is a research scholar in the Department of Computer Science, Bishop Heber College (Autonomous). He was the assistant professor at Don Bosco College of Arts and Science, Keela Eral, Tamilnadu. His area of interest is Internet of Things.

Dr. Jemima Priyadarsini R., is the associate professor in the Department of Computer Science, Bishop Heber College (Autonomous). She has been teaching here for the last nineteen years. Her research areas are Cloud computing, Big Data and Internet of Things. She has published more than 25 articles in international journals.

How to cite this article:

S. Alexander Suresh, R. Jemima Priyadarsini, "Design of Maintaining Data Security on IoT Data Transferred Through IoT Gateway System to Cloud Storage", International Journal of Computer Networks and Applications (IJCNA), 9(1), PP: 135-149, 2022, DOI: 10.22247/ijcna/2022/211632.