Journal of Theoretical and Applied Information Technology
15th November 2022. Vol.100. No 21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

LIGHTWEIGHT IOT IMAGE ENCRYPTION

HANEEN DWEIK [1], MOHAMMED ABUTAHA [2], RACHID SAMMOUDA[3]

College Of Information Technology And Computer Engineering [1, 2], Associate Professor Of Computer Science [3]
Palestine Polytechnic University [1, 2], King Saud University [3]
E-mail: 196001@ppu.edu.ps [1], m_abutaha@ppu.edu [2], rsammouda@ksu.edu.sa [3]

ABSTRACT

Nowadays, all smartphones, laptops, and other communication devices connect to the cloud, making data accessible to everyone. The Internet of Things (IoT) network is a group of various devices interconnected over the internet that exchange data between themselves and other services. IoT has a wide application range from smart applications to a variety of industrial applications. Because nodes in an IoT network have limited resources, classical cryptography methods are costly and inefficient, so lightweight block ciphers are one of the most sophisticated ways to overcome security shortcomings in this environment. The result is a low bandwidth, reduced storage space, and shortened computation times due to the compression. Specifically, this paper discusses the performance of lightweight AES algorithms for encrypting and decrypting images in smart IoT devices.

Keywords: - *Internet of Things (IoT), Lightweight, Image, AES, Blind people.*

1. INTRODUCTION

In the field of cutting-edge remote media communication, IoT describes physical objects (or groups of such objects) that have integrated sensors, processing ability, software, and other technologies to connect and exchange data with other devices and systems over the Internet. It is quickly establishing itself as a new paradigm. In the IoT, people, data, processes, and things are connected to make network connections that are more relevant and useful than ever before. With the rapid advancement of IoT, it is exposed to numerous risks and challenges, such as handling huge amounts of data, processing energy efficiently, responding to security threats, and encrypting/decrypting huge amounts of data. The concept refers to a system of interlinked computing items, such as Radio-frequency identification (RFID) tags, sensors, actuators, and cell phones; digital machines; and people, allowing the sharing of data over a network without the need for human-to-human interactions. In an IoT world, massive amounts of raw data will be continuously collected, requiring real-time sensor data streams as well as techniques for converting this raw data into useful information. Furthermore, data privacy and security will be a serious concern. A cryptographic algorithm designed for a device with incredibly low resources will have different design criteria than one commonly used. Modern cryptography has evolved from this very specific area into lightweight cryptography. The low energy requirement of these algorithms makes them resistant to physical attacks. The storage space requirements of multimedia applications are more challenging due to the size of multimedia data, the need for real-time processing, transmission delay, and security protection. Many new applications have emerged in the IoT and cloud computing fields, where multiple devices and servers perform thousands of operations at the same time. Multimedia applications require real-time processing, resulting in a critical role for encryption and decryption speed. A variety of technological fields, including smart cities and homes, have benefited from the Internet of Multimedia Things (IoMT). Most multimedia contents require large storage discs to be uploaded and streamed to different devices. As a result, video data or media content can be thoroughly analyzed if an issue occurs. IoT devices are low-powered and small in size, so they need a cloud platform or third-party storage device to store, operate, and process information collected.

A majority of encryption is related to encrypting and decrypting text messages or documents, but images are also a prime bearer of crucial information, therefore, they need to be encrypted. The encryption process involves modifying the pixels of an image so they are no longer representative of the original image. Once the receiver receives the encrypted image, it must be decrypted to reconstruct the image. Having encrypted images ensures that even if an interceptor gets access to a picture during

transmission they are incomprehensible to them. Another practical use of encrypted images is for the security of biometric data. Fingerprint and retina scans involving biometric identification have become increasingly common, so these data must be securely shared and stored. When data is encrypted, it can be unintelligible to the intruder even if it is accessed maliciously.

IoT security has been emphasized by many organizations and research agencies. Open Web Application Security Project has identified privacy issues, inadequate authentication/authorization, lack of transport encryption, and poor physical layer security as the main causes of cyber-attacks on IoT. Identifying a device, validating its identity, authorizing it, establishing keys and managing them, as well as establishing trust and reputation are the five features in IoT security. Cryptographic primitives can help accomplish all of these objectives, including authentication, access control, non-repudiation, confidentiality, integrity, and availability.

This paper discusses the performance of lightweight AES algorithms for encrypting and decrypting images in smart IoT devices. The rest of the paper is organized in the following manner: in Section 2, we have presented a literature review. In Section 3, we have discussed the Advanced Encryption Standard (AES) and have described its detailed architecture framework. Section 4 presents the Internet Of Things. Section 5 presents the compared the current algorithms and approaches. Section 6 presents the results and discussion, while we have discussed the conclusions and provided further suggestions.

2. LITERATURE REVIEW

Several systems and approaches are proposed to address the challenges and restrictions involved with the encrypted transmission of big multimedia data. Moreover, the security of multimedia data needs to be researched further. This section presents studies that have been previously conducted in different categories.

In their study, Aljawarneh, et al. [2], the encryption of big multimedia data, developed and designed a multithreaded encryption algorithm system. An advanced encryption standard (AES), genetic algorithms, and the Feistel Encryption Scheme (FEES) have been used in this system. The system was evaluated concerning computational run time and throughput for the encryption and decryption process to analyze the performance of the system on actual medical data and benchmarked against the RC6, MARS, 3-DES, DES, and Blowfish algorithms. They have been implemented the proposed encryption system with a multithreaded programming approach to improve efficiency and performance. Finally, they have been tested their system against the sequential version to evaluate its resource efficiency. Comparing our system to other available encryption algorithms, their results showed that our system took the least amount of time to run and delivered a higher throughput. Furthermore, they were also able to achieve a 75% improvement in computation run time and a 4-fold increase in throughput versus their sequentially structured version. Based on the security objectives, the algorithm performed better than existing algorithms in achieving the Avalanche Effect which is the minor changes in the plaintext lead to completely different output when the cipher algorithm is used, and they can therefore include it in any encryption/decryption process of large, plain, multimedia data.

GUEAIEB, W. AND MIAH, M. S. [3], blind people need assistance in detecting obstacles, finding locations, and getting directions while moving around to reach their specified destination. Based on this persistent need, we suggest a navigation system to facilitate these requirements. We provided this navigation system to blind students, employees, or guests within King Saud University campus area. The technologies used in our proposed system are: the blind mobile device, RFID tags and Reader, GPS, text to speech, voice Recognition, and WiFi. The system detects the blind location using GPS, if internet connection is available, and uses RFID tags fixed outdoors and indoors on the building in the path, WiFi routers are used indoors to detect the location. The system uses voice recognition and text to speech to communicate with the blind to lead him to his destination and to give him the directions. The results show good performance in obstacles avoidance and in blind guidance.

Zhang [4], proposed an image cryptosystem that is compared with existing chaos-based image cryptosystems based on encryption/ decryption speed and security performance. In simulations, AES was shown to apply to image encryption, which argues against the commonly held perception that AES is not suited to image encryption. As a result of this paper, it has been recommended to use AES-based image encryption as a benchmark for the speed of image encryption algorithms. All other encryption algorithms whose speeds are lower should be discarded in practical communications.

Authors in [5], according to their study, AES cannot cryptograph images in CBC mode. However, AES in CBC mode could be used to encrypt images. AES can be used to encrypt an image and generate an initial vector (IV). AES is secured by far, so the

tested image cryptosystem is secure. Simulation results indicate the AES-based image cryptosystem is faster than some chaotic systems-based image cryptosystems. The tested system can thus be used as a reference for comparing other newly offered image cryptosystems. Cryptosystems for images that perform encryption and decryption slower than AES in the same computer need to be enhanced.

Rana, et al. [6], proposed a lightweight cryptographic algorithm with 16.73% lower power usage than the existing cipher. Modern electronics and the internet will enable resource-constrained devices to become daily necessities for everyone, so data security will be an important consideration. Those devices will be communicating with one another incessantly, so information must be protected at all times. The implementation shows promising performance making the algorithm an ideal candidate for resource-constrained devices.

Lal Chowdhary, et al. [7], proposed an analysis to decrypt and encrypt images using hybridization of Elliptic Curve Cryptography (ECC) and Hill Cipher (HC), ECC and AES (Advanced Encryption Standard), and ElGamal and Double Playfair Cipher (DPC). The measurements used in this analysis are (i) encode and decrypt times, (ii) entropy of the encrypted image, (iii) intensity loss of the decrypted image, (iv) Peak Signal to Noise Ratio (PSNR), (v) Number of Pixel Change Rate (NPCR), and (vi)Unified Average Changing Intensity (UACI). ECC and ElGamal cryptosystems offer asymmetric key cryptography, while HC, AES, and DPC provide symmetric key cryptography. Hybrid processes combine the speed and ease of implementation of symmetric algorithms with the security of asymmetric algorithms. According to the metric measurement with test cases, ECC and HC have a good overall solution for image encryption with smaller image sizes when using AES with ECC.

Ji, et al. [8], proposed an improved AES-ECC hybrid encryption system that has good flexibility and versatility and optimized ECC multiplication unit design according to the characteristics of wireless sensor networks. It was capable of generating and authenticating digital signatures at a faster rate. It also fully met wireless sensor networks' reliability, processing power, and power consumption requirements. AES encryption module is currently undergoing high-performance enhancements (increase throughput, decrease logic unit occupancy) and optimizations of ECC cryptographic module random point multiplications are currently being implemented. There are three properties of the proposed scheme: (1) it provides better security with relatively low resource requirements, (2) it is straightforward to administer keys, and (3) it is resistant to some attacks and a digital signature can be generated and verified quickly and easily.

Jolfaei(B), et al. [9], proposed a method that encrypts texture images via bit masking and permutation procedures using Salsa20/12 stream cipher as part of a novel texture encryption scheme that complements the existing methods for 3D object encryption. As a result, the method has very low overhead and meets the security requirements, and protects the 3D surface geometry from partial disclosure by keeping the texture patterns hidden. Compared to full encryption and selective encryption (using the 4 most significant bits), the scheme has a higher speed-security profile. The schemes are implemented and tested with 500 sample texture images. Comparing the experimental results with full/selective encryption by 128-bit AES, the scheme demonstrated better encryption performance.

M.Sankari P. Ranjana [10], proposed protecting the image data in the mobile cloud through privacy-preserve lightweight image encryption (PLIE), they have been introduced a method that keeps metadata on mobile while maintaining user privacy. Mobile data is split, distributed, and scrambled (SDS) to maintain user privacy and store it in the cloud. As a result, the throughput increases, the encryption time is sped-up, and the complexity is minimized. Using the PLIE method implemented in Python language, the encryption time was approximately 50% shorter than that of AES. They have been measured the performance of the existing method (AES) versus the proposed method (PLIE) using various parameters. Furthermore, they have been evaluated the security level by presenting some security attacks.

3. ADVANCED ENCRYPTION STANDARD (AES)

The AES encryption algorithm is symmetric in the group, and there are three different key lengths: 128 bits, 196 bits, and 256 bits, with the packet size being 128 bits. The algorithm is reasonably flexible in its application. The AES algorithm is widely used in software and hardware. In the three key lengths, the 128bit key length is commonly used. The internal algorithm performs a ten-time iterative process when the key length is under. The five sections of the final round are joined by the Sub Bytes, S-box, Shift Rows, Mix Columns, and Add Round Key. AES has five different units of measurement: bits, bytes, characters, groups, states. A round of AES is composed of byte replacement (Sub Bytes), line displacement (Shift Rows), mixed column

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

displacement (Mix Columns), key replacement (Add Round Key), and so on. AES algorithm design should meet three criteria during all phases of the data packet transformation, in the beginning, and ending stages of encryption:

(1) Can resist all known attacks.
(2) Fast and coding compaction.
(3) Simple in design.

Fig.1. shows the process of AES Encryption and Decryption. It relies on the packet size and the length of the key, and it is controlled by the key. The iteration round of the number is controlled by the key and the length of the block.

As a Fig.1, a cryptographic algorithm is shown on the left and a cryptographic algorithm is shown on the right of the figure. A key expansion algorithm is shown in the middle of the figure. It consists of N iterations having four different steps: byte replacements (Sub Bytes), line displacements (Shift Rows), mixed column shifts (Mix Columns), and key shifts (Add Round Key). There are no mixed column transformations in the final round. The decryption algorithm is the opposite of encryption (inverse byte substitution, inverse shift rows, and

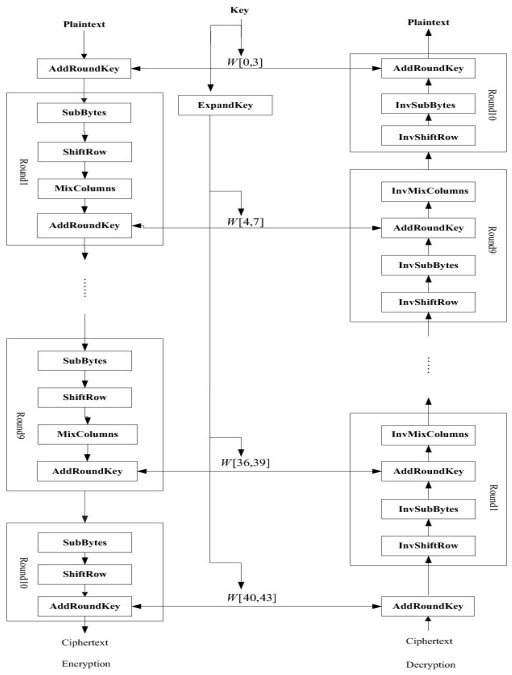


Figure 1: Process of AES Encryption and Decryption [11]

inverse mix columns).

For full encryption, the data is passed through Nr rounds (Nr = 10, 12, 14). These rounds are governed by the following transformations:

- **Sub byte Transformation:** A non-linear substation table is used (s-box). It's constructed by multiplying inverse and affine transformation.
- **Shift rows transformation**: The offset of the left shift varies between one and three bytes, and the last three rows of the state are cyclically shifted.
- **Mix columns transformation:** The result is equivalent to multiplying columns of the states by a fixed matrix for each column vector. Note that the bytes are treated as polynomials rather than numbers.
- **Add round key transformation:** The round key is XORed with the working state, which is its overexpansion **key:** Even if an eavesdropper knows the plaintext and cipher text AES algorithm can't be determined because the secret key is known to both the sender and the receiver. According to its specifications, AES uses one of three key sizes (Nk). AES-126, AES-196, and AES-256 respectively use 128 bit (16 bytes, 4 words) and 196 bit (24 bytes, 6 words) key sizes. Key values have no weak point, unlike DES. All key values are equally secure, therefore no key-value renders encryption more vulnerable than the other. Key values are expanded via key expansion routines before being used in the AES algorithm. In addition to performing "on the fly" word expansion, this routine can be performed at any time.

3.1. AES key expansion

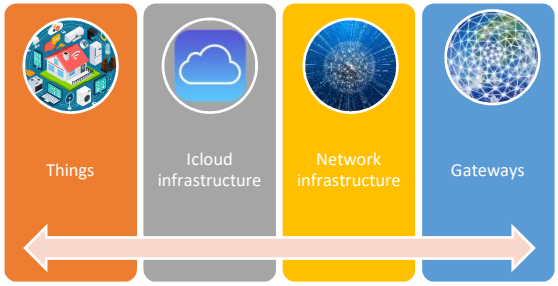
Add Round Key transformation uses a sub-key for every round, which corresponds to the number of bytes from the initial key. For example, AES-128 converts to 44 bytes per word, and every word are indexed as W[index] = [0... 43]. The first set of columns (W0, W1, W2, and W3) are all full with the given cipher key and the columns in locations that are multiples of four (W4, W8, W12... W40) are all generated using the following three operations:

- Rot Word: Rot Word rotates a word to the left for one rotation.
- Sub Word: With Sub Word, individual bytes are replaced.

- Word Wi-4 and a defined constant from the Recon matrix are XOR'd with the result of Rot Word and Sub Word operation

4. INTERNET OF THINGS (IOT)

Internet of Things refers to a paradigm that encompasses the considerable presence of various objects that can communicate with one another over



wireless and wired connections. The rapid evolution of the classical Internet into the IoT enables the exploration of a multitude of previously unimaginable domains of utility. Then, an asymmetric secure pipeline is created between the communicating entities in order to ensure the security of the exchanged data. Mutual authentication is necessary between a given device and its device manager. As it develops rapidly, IoT is criticized for not considering the significant security challenges it is bringing along with the regulatory changes it demands.
The security of IoT must be addressed by first understanding all the components that makeup IoT.

Figure 2: IOT Architecture

Next, each component must be identified and the weakness of each area needs to be countered with the appropriate technologies. As shown in Fig.2.the IoT architecture consists of things, gateways, network infrastructure, and cloud infrastructure.

5. COMPARED THE CURRENT ALGORITHMS AND APPROACHES

In [17],[8], [9], and [10], by comparing three papers, we presented them in our study, which compared their results.

5.1. Boolean functions

S-boxes (substitution boxes) is a basic component of symmetric key algorithms that perform substitutions. They are used to draw symmetric key algorithms because of their properties. Boolean functions on n variables have the form F_2^n-F_2. Therefore, it is possible to define S-boxes as (n, m) Boolean functions. A good cryptographic S-box will have properties such as nonlinearity, bijection, the strict avalanche criterion, the output bits independence criterion (BIC), and an equiprobable input/output distribution of XOR bits.

5.1.1. NIST statistical test suite

Cryptographers use the 15 NIST tests to assess the randomness of binary sequences. These tests evaluate the various attributes of a sequence with respect to its non-randomness. To determine randomness, NIST's Statistical Test Suite will be used to test the S-boxes generated, the chaotic PRNG, as well as the ciphered images.

5.1.2. The Lorenz System

Chaos is characterized by a strong sensitivity to initial states. The Lyapunov exponent provides a quantitative description of this sensitivity. Lorenz maps exhibit similar chaotic behavior.

$$\begin{cases} \dot{x} = a(y - x) \\ \dot{y} = cx - y - xz \\ \dot{z} = xy - bz \end{cases} \quad (1)$$

Fig. 3 and 4 show the Lorenz attractor and the Lyapunov exponent, respectively.

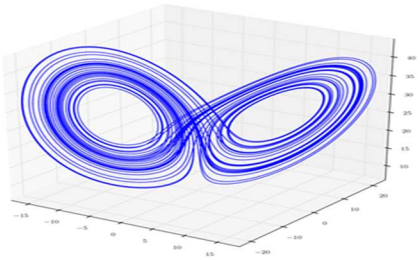


Figure 3: The Lorenz attractor [17]

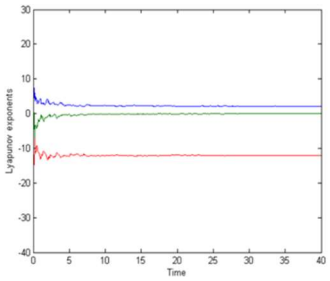


Figure 4: The lyapunov exponents. [17]

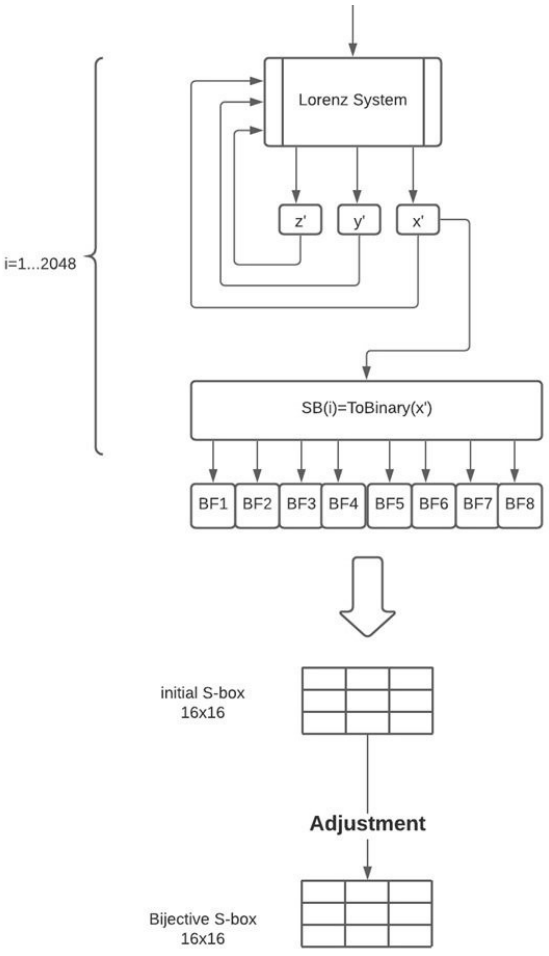


Figure 5: Flowchart of the initial S-box. [17]

To create Boolean Functions, the Lorenz system generates chaotic binary sequences.

5.1.3. Modified AES S-Box Generation [17]

It is implemented as a Lookup Table (LUT) that transforms m input bits into n output bits, where n can differ from m. This is known as the (m*n) S-box. This device is designed to be resistant to linear and differential cryptanalysis, and it transforms one byte of input data into another.

Based on a chaotic map, the S-box generating algorithm generates a 2048-bit set S. Then, 8 subsets of Boolean functions from S are constructed. Fig.5 shows the construction process.

5.1.4 IMAGE ENCRYPTION SCHEME [17]

Lightweight cryptography must consider many factors and constraints on sensor nodes; essentially, power, memory, computation speed, and bandwidth. Since the execution time is affected by the speed, power consumption is a function of the number of computations that determine the processing speed. Thus, the number of computations influences the processing speed, which is then a function of lightness.

Security is achieved via lightweight encryption, which is a method of overall system security used in this paper. To be secure, the proposed cryptosystem needs to be based on a modern algorithm that shows a sufficiently high level of security. A number of optimized implementations have been suggested in the context of lightweight algorithms that improve their performance.

When implementing an encryption scheme in a wireless sensor, the memory size and energy consumption are determining factors. A device with a RAM size of 8 KB and a ROM size of 116 KB cannot perform the following actions simultaneously:

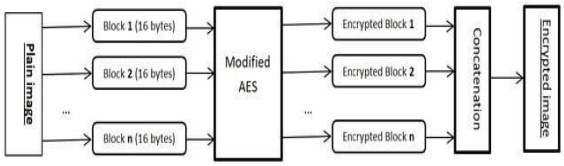


Figure 6: The flowchart of the main algorithm [17]

- Store the Tiny OS operating system,
-Store the encryption algorithm,
-Store the grayscale image to be encrypted and
-Run the algorithm in order to generate the encrypted image.

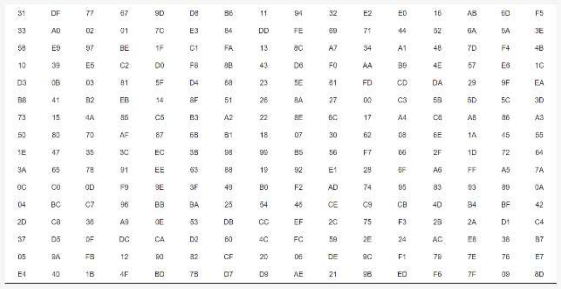
In order to solve this problem, they have been developed a lightweight algorithm and implemented it on the XM1000 wireless sensor. They have been demonstrated that the algorithm can encrypt large

Journal of Theoretical and Applied Information Technology
15th November 2022. Vol.100. No 21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

grayscale images. Fig.6. shows the flowchart of the main algorithm

In order to implement the algorithm, they have been proposed that the image be encrypted by blocks of 16 bytes. After running the code with larger blocks, they have been observed that the sensor chose to



produce non-standard results, which is why they have been chosen to use 16 bytes. Using the Boot.booted() event, the main code encrypts the grayscale image. To get the ciphered image from the sensor, they have been divided it into blocks of 4 * 16 bytes and send them separately. With TinyOS, Timer.startPeriodic() sets the period, and Timer.fired() sends the ciphered blocks. For image size, they have been used 50 * 64 bytes, and the period was 5 *10*2 s. A ciphered image is generated from all the blocks in the received sequence. In Table 1, the generated S-box is shown.

Table 1: THE S-BOX

5.2. Texture Encryption Scheme [9]

In [9], Using Salsa20/12 for the upper nibble image encryption, the bitstreams of the lower nibble image are scrambled by permuting a zigzag pattern on the bitstream. They call our encryption mechanism 'Salsa Dance' since it is consistent with (Latin American) Salsa movement. Infer the steps of the encryption algorithm, P being the plain image, N being the nibble image, and C being the cipher image. If RGB is a 24-bit representation, each flat image, nibble image, or cipher image is represented by three M * N matrices, so R, G, and B color layers. For any pair of x ($1 \le x \le M$) and y ($1 \le y \le N$), multiply p (x, y) by n (x, y) and the result is the flat image, nibble image, or cipher image. P (x, y) and c (x, y) are the entry values for the plain-images, nibble images, and cipher images, respectively; n (x, y) $\in \{0, 1, \cdots, 15\}$.

A 24-bit texture image with one color layer has the encryption procedure described below. For the other color layers, the procedure is similar. By splitting every entry into upper and lower nibbles, we can break the plain image into two nibble images that correspond to x and y. For any x ($1 \le x \le M$) and y ($1 \le y \le N$), n1 (x, y) and n2 (x, y) are defined as follows:

$$n_1(x, y) = p(x, y) \bmod 2^4, \quad (2)$$

$$n_2(x, y) = (p(x, y) - n1(x, y)) \cdot 2^{-4}. \quad (3)$$

Fig.7. shows a zigzag path for scanning an image with the dimensions 3 * 4 in Fig.7a if mod(s, 12) = 7. In this case, entry scanning starts at the 7th entry

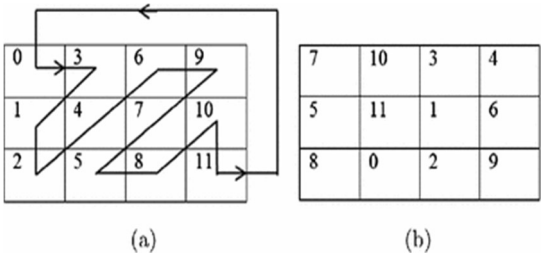


Figure 7: (a) A zigzag path to scramble bits of a bit-plane image,

and ends at the 9th entry, which is the entry immediately before the initial one. Scanning involves the placement of bits, column by column, in a matrix sequentially as they are encountered. Permutation affects both bit-plane image bits (diffusion) as well as the values of nibbles (confusion). A mod (s, 12) = 7, the permutation result of the test bit-plane image can be seen in Fig. 7b. Following the permutation process, when the scrambled bit-plane image is combined with every 4 consecutive columns, the encrypted lower nibble-image with size M * N can be reconstructed.

The final step is to create the cipher image by combining the encrypted upper and lower nibble images. To summarize, the whole encryption process is as follows:

$$P = 2^4 \cdot N_2 + N_1, \quad (4)$$

$$C = E(P) = 2^4 \cdot E_2(N_2) + E_1(N_1), \quad (5)$$

Where,

$$E_2(N_2) = \text{Salsa20/12}(N_2) \quad (6)$$

$$E_1(N_1) = \text{Perm}(N_1) \quad (7)$$

PK (plain image), N1, N2 (lower nibble image), and C (upper nibble image) refer to plain, lower, and upper nibble images, respectively. In decryption, cipher images are further divided into upper and lower nibble images. In 24-bit texture images, there is a close correlation between different layers of color in the image. The upper and lower nibbles are decrypted with the same keystream used in encryption, while the inner nibble is decrypted by inverse permutation. To meet this requirement,

Salsa20/12 uses a 64-bit nonce each time the color layer is encrypted. The same message will never be encrypted twice in the same way so that there is always a different ciphertext. If the same nonce and key are used on two different plaintexts, then you can cancel the keystream out by masking the ciphertexts together.

They have been evaluated the performance of the proposed cipher with: (i) the full encryption using 128-bit AES, (ii) the selective encryption of four of the most significant bit planes using 128-bit AES, and (iii) Salsa Dance, using a processor and memory combined with an Intel Core 2 2.4 GHz. The texture images have been encrypted with the ECB mode of the AES algorithm. AES provides several modes of operation, among which ECB allows simultaneous encryption/decryption and achieves better performance by using a simple sequential message schedule. CGT textures provided a test image with its corresponding encryption results. Fig.8 shows one test image with its corresponding encryption results. The algorithm is also suitable for applications that require random access to encrypted data. AES encryption cannot annihilate the coarse patterns of the texture image, whereas Salsa Dance dissipates the correlation among the entries of the texture image.

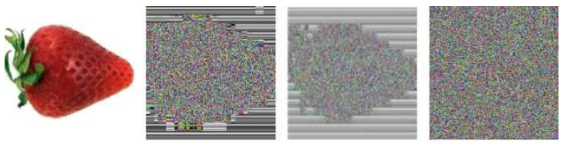


Figure 8: Encryption results of a sample texture image:

(a) Original image, (b) encrypted [9]

A pseudorandom binary sequence of size 12MN + 3[$\log_2$ (4MN)] is used to encrypt 24-bit texture images with MN bits (i.e., 24 MN bits in total). This means that the proposed method reduces the computational cost by approximately half compared to conventional full encryption methods. The proposed cipher encrypts input data by creating a pseudorandom sequence that covers almost 50% of the input data. Therefore, reducing the computational costs can result in savings in computing power, storage, processing time, as well as transmission bandwidth, which enabled more processes to be run simultaneously.

Results of timing tests suggest that the 4 bit-plane selective encryption methods are slower than Salsa Dance by an average of 247 %. The AES full schemes are slower by an average of 495 %. Experimentally, Salsa Dance shows that provide a better level of encryption than both full and selective encryption with AES.

5.3. Plie method [10]

In [10], this method ensures image data security using three different processes, of splitting, distributing, and scrambling the images. In addition, it ensures user privacy by keeping metadata in the mobile device, and finally storing it on the cloud. A split image file is broken into two parts: the header and the contents. The header contains several privacy-protecting features, including the image type, size, date of creation, chunk size, height, width, and resolution. There are many chunks of content. For distribution, chunks may be divided based on a pattern, such as a key, a predefined function, or an individual chunk. PLIE categorizes patterns as odd chunks (file1) and even chunks (file2) that are sequentially repeated. The maximum number of chunks is m= (image size/chunk size) - header size, where the image size is the size in bytes of the image file, and chunk size is the size of the chunks. The proposed PLIE method reduces encryption time by 50% approximately compared to AES.

Python has been used to implement PPIE. The performance of PPIE is evaluated on metrics such as throughput, minimize the encryption time, Key sensitivity, and low complexity to maintain privacy.

5.3.1. Plie algorithm plie algorithm [10]

Input:

The matrix of the image 'I', number of pixel in image 'n', maximum number of chunks formed 'm'

Output:

The encrypted image E (I), key1, key2, H (i)

Process:

Generate image file 'I'

Convert image file into binary data B (I)

Split:

For each pixel at position (i, n-1) in I do

For each pixel at position (k,m) do Split the image as header

H (i) and content

C (i,k) B(I)← H(i)+C(i,k);

End for

End for

Distribute (pattern):

Distribute C (i,k) as different file by grouping of chunks based on pattern

File1←Collection of even chunks(C (i,k[even])

File2←Collection of odd chunks(C(i,k[odd])

Scramble: Generate key1 as first row of the file1 and key2 as second row of the file2 Scramble file1 by key1 and file2 by key2
Such as file1←key1+file1
file2←key2+file2 E (I) ←encrypt (file1 and file2)
Return E (I), key1, key2, H (I)

5.3.2. Decryption algorithm [10]

The original image is regained from the reverse of the encryption process.

Input:
Encrypted Image E(I),key k1,key k2,Header of the file H(I)
Output:
The original image 'I'
Process:
Fetch E(I)
Decrypt(E(I)) from cloud
Collection of file1 and file2
Compute key k1,key k2←mobile
file1←file1-key k1
file2←file2-key k2
Collection of chunks C[i,k]←file1 &file2
B(I)← H(I)+C[I,k];[Header of the file as H(I)]
Convert B(I) to I
return I

6. RESULTS

After studying the previous papers, we will compare by showing the different performance results. Table.2 shows the different results:

6.1. PERFORMANCE ANALYSIS

Table 2: Comparison of Performance Analysis [8], [9], [10], [17]

Performance Analysis	Size of input image encrypt	output image decrypt	Algorithm used	Encryption Time (seconds)	Decryption Time (seconds)
In [8], Eggs (Grayscale 256 × 256 Pixels) Image	256 × 256 pixels	256 × 256 pixels	AES-256 with ECC	2.82401	2.75127
In [8], Eggs (colored 256 × 256 Pixels) Image	256 × 256 pixels	256 × 256 pixels	AES-256 with ECC	2.52632	2.50829
In [8], Mona Lisa (Grayscale 256 × 256 Pixels) Image	256 × 256 pixels	256 × 256 pixels	AES-256 with ECC	2.84150	2.7866
In [8], Mona Lisa (Colored 256 × 256 Pixels) Image	256 × 256 pixels	256 × 256 pixels	AES -256 with ECC	2.53642	2.52728
In [9], Selective AES	M × 4N	M × 4N	ECB mode of AES-128	2.47	-
In [9], Full AES	M × 4N	M × 4N	ECB mode of AES-128	4.95	-
In [9], (Salsa Dance)	M × 4N	M × 4N	ECB mode of AES-128	1.00	-
In [10], Baby	256 × 256 pixels. That used file sized 4.9 KB	-	AES-128	0.0007	-
In [10], Leaf	File size 5.7 KB	-	AES-128	0.0009	-
In [10], Wheel	File size 6.5 KB	-	AES-128	0.00113	-
In [10], Ball	File size 8.7 KB	-	AES-128	0.00113	-

In [17], they have been chosen solution was to implement it on the XM1000 sensor, which is composed of a MSP430 microcontroller and CC2420 radio chip. TinyOS is an embedded operating system written in nesC. 230,399 milliseconds is the execution time of the algorithm under a physical sensor (XM1000 sensor), not a simulation, which resulted in real results. They have been ciphered an image of 50 * 64 bytes, consuming 13,624 KB in ROM and 7826 KB in RAM.

6.2. Security

The following parameters were considered in the comparative analysis: ECC with AES, test samples [7].

- **Metric Values for PSNR (dB): 10 dB** expected value for the reconstructed image. It measures the signal-to-noise ratio in decibels between two images. It is used to determine whether the original image is better than the compressed image. For the egg (grayscale), egg (colored), Mona Lisa (grayscale), and Mona Lisa (colored), the PSNR values are between 8 and 9.5.
- **The NPCR metric value (100%)** is the expected change in the cipher image's pixels (when only one pixel of the plain image is changed) when the number of pixels from the input image is varied in the encrypted image. Based on the result of the Eggs (Grayscale), Eggs (Colored), Mona Lisa (Grayscale), and Mona Lisa (Colored) tests, it indicates that there is a significant number of pixels that differ from the original image in the encrypted image.
- **The average value of the UACI (30%)** is for varying numbers of pixels in the encrypted image from the input image. The UACI measure shows how secure an algorithm is against differential attacks, such as plaintext attacks or cipher-only attacks. Higher values indicate that this image is more resistant to such attacks. Values obtained for Eggs (Grayscale), Eggs (Colored), Mona Lisa (Grayscale), and Mona Lisa (Colored) range from 26% to 30%.

- **Square error**: in a decrypted image, the square error represents the discrepancy between the decrypted image pixels and the original pixels. For a good algorithm, the square error should be close to **zero.**

Table 3 shows the metric measures for security.

Table 3: Metric measures

Evaluation Metrics For	Squared Error in decrypted image	PSNR (10 dB)	NPCR (100%)	UACI (30%)
Eggs (grayscale)	↑	↑	↑	↓
Eggs (colored)	↑	↓	↑	↑
Mona Lisa (grayscale)	↑	↓	↑	↑
Mona Lisa (colored)	↑	↑	↑	↑

↓↑: The arrows show the comparison between metrics measures for many images.
Based on the test image in Fig. 8.a, Table 4 offers the PSNR values for the encrypted images. Given the test image in Fig. 8 a, it is apparent that encryption using slightly different secret keys results in different Salsa Dance or AES cipher images. Salsa Dance, however, generates more dissimilar cipher-images than selective/full AES although the secret key is only changed by one bit. Thus, the proposed method is highly sensitive to changes in the key, making the adversary's analysis of Salsa Dance even harder in terms of finding any relationship between the keys used.

Table 4: Comparison of the PSNR values

Encryption schemes	Selective AES	Full AES	Proposed (Salsa Dance)
Between the original and encrypted image with the original key	↓	↓	↑
Between the original and encrypted image with 1-bit different key	↑	↓	↓
Between the encrypted images using the original and modified keys	↑	↑	↓

6.3. Lightweight algorithm

Table 5 shows the algorithms used in the three papers that were compared.

Table 5: Lightweight Algorithm used

Compared	Lightweight Algorithm
In [9]	Salsa Dance
In [10]	PLIE Method
In [17]	Modified AES

Salsa Dance encrypts textures with bit masking and a permutation procedure employing the Salsa20/12 stream cipher, which is relatively lightweight compared to traditional 128-bit AES encryption. In addition, Salsa Dance reduces the correlation among the entries in the texture image, thereby diluting the coarse pattern of the plain image and preventing any data leakage. According to the key sensitivity analysis, slight changes in the secret key can result in entirely different cipher-images. Therefore, even if the encryption and decryption keys differ slightly, the original image cannot be recovered. Salsa Dance conceals the shape and boundaries of the underlying 3D object while AES encryption can be detected by edge detection, even when using full and selective encryption. Salsa Dance's texture encryption not only protects texture images from most surface reconstruction attacks but also ensures the security of protected 3D models [9].
For image encryption in mobile, PLIE method reduces resource usage, throughput, speed-up processing time, and has a reduced complexity. Through different performance measurements, we've proven that we can keep user privacy while reducing encryption time by half compared to existing methods like AES [10].
Using chaotic Boolean functions, a lightweight encryption algorithm was based on the standard AES. Permutation and diffusion phases were accomplished using Hilbert curve scan patterns and Lorenz systems. Cryptographic properties of the chaotic S-box and NIST tests validate its power. A prototype of the algorithm was developed to be implemented in low-power IoT devices, and a grayscale image was encoded with an XM1000 sensor to demonstrate its effectiveness. Since the algorithm is light enough, it satisfies many criteria, such as memory consumption, execution time, and information entropy [17].

6.4. Discussion and Comparative Analysis

According to the first paper [8], compression of the image uses techniques that use less space to provide the same information, which solves the computation and high protection problem. The result is a low

bandwidth, reduced storage space, and shortened computation times due to the compression.

According to the second paper [9], that describes a technical solution for meeting the confidentiality requirements associated with texture images that overcome the limitations of current techniques, in addition, large data volumes and high application requirements, including real-time performance, complexity, and security, are common.

According to the third paper [10], to reduce resource consumption, throughput, increase processing speed and reduce complexity, the PLIE method is an excellent choice for image encryption on mobile devices, It has been shown by a variety of performance measurements to maintain privacy for users in mobile and to reduce encryption time by nearly 50% compared to existing methods such as AES.

For the study, input and output images include Mona Lisa (Grayscale 256 * 256 Pixels), Mona Lisa (Colored 256 * 256 Pixels), and Eggs (Grayscale 256 * 256 Pixels). Representative input and output images, with proposed encryption and decryption algorithms, are provided [8].Fig.9. Shows Sample input and output for proposed hybrid algorithms.

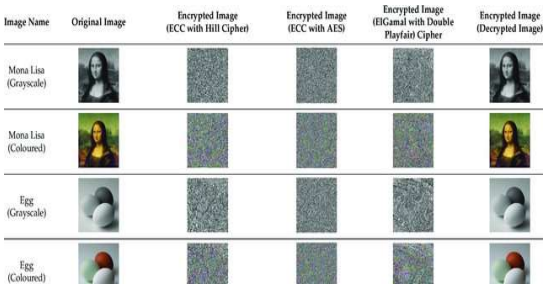


Figure 9: Sample input and output for proposed hybrid algorithms. [8]

When we compared the proposed systems, we found that these modifications were made to the original AES algorithm, while the original algorithm security remains robust, the modified AES algorithm remains lightweight and faster, providing more satisfaction for embedding in IoT devices and sensors that consume little power.

7. CONCLUSION

Nowadays, all smartphones, laptops, and other communication devices connect to the cloud, making data accessible to everyone. IoT network is a group of various devices interconnected over the internet that exchange data between themselves and other services. It has a wide application range from smart applications to a variety of industrial applications. Encryption is one of the best techniques to guarantee end-to-end security in the IoT network, as the volume of data transferred is so high. Because nodes in an IoT network have limited resources, classical cryptography methods are costly and inefficient, so lightweight block ciphers are one of the most sophisticated ways to overcome security shortcomings in this environment.

When we compared the proposed systems, we found that these modifications were made to the original AS algorithm, while the original algorithm security remains robust, the modified AES algorithm remains lightweight and faster, providing more satisfaction for embedding in IoT devices and sensors that consume little power.

In future work, we will work on lightweight AES algorithms and chaotic to ensure more security and speed in the image encryption process.

8. ACKNOWLEDGMENT:

This project was funded by the National Plan for Science, Technology and Innovation (MAARIFAH), King Abdulaziz City for Science and Technology, Kingdom of Saudi Arabia, Award no. INF2696-02-12.

REFERENCES:

[1]- PDFprof.com. 2022. *ciaa information security Information Security PDF | PDFprof.com*. [online] Available at: <https://pdfprof.com/EN/PDF_Documents_Doc.php?q=3PDF48588-ciaa+information+security> [Accessed 2 January 2022].

[2]- Aljawarneh, S., Yassein, M. and Talafha, W., 2017. A multithreaded programming approach for multimedia big data: encryption system. *Multimedia Tools and Applications*, 77(9), pp.10997-11016.

[3]- Gueaieb, W. and Miah, M., 2009. A Modular Cost-Effective Mobile Robot Navigation System Using RFID Technology. Journal of Communications, 4(2).

[4]- Zhang, Y., 2018. Test and Verification of AES Used for Image Encryption. *3D Research*, 9(1).

[5]- Zhang, Y., Li, X. and Hou, W., 2017, June. A fast image encryption scheme based on AES. In *2017 2nd International Conference on*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Image, Vision, and Computing (ICIVC) (pp. 624-628). IEEE.

[6]- Rana, S., Hossain, S., Imam, H. and Mohammod, D., 2018. An Effective Lightweight Cryptographic Algorithm to Secure Resource-Constrained Devices. *International Journal of Advanced Computer Science and Applications*, 9(11).

[7]- Chowdhary, C., Patel, P., Kathrotia, K., Attique, M., Perumal, K., and Ijaz, M., 2,020. Analytical Study of Hybrid Techniques for Image Encryption and Decryption. *Sensors*, 20(18), p.5162.

[8]- Ji, B., Wang, L. and Yang, Q., 2014. New Version of AES-ECC Encryption System Based on FPGA in WSNs. *Journal of Software Engineering*, 9(1), pp.87-95.

[9]- Jolfaei, A., Wu, X. and Muthukkumarasamy, V., 2016. A Secure Lightweight Texture Encryption Scheme. *Image and Video Technology – PSIVT 2015 Workshops*, pp.344-356.

[10]- Sankari, M. and Ranjana, P., 2018. PLIE- A Light-weight Image Encryption for data Privacy in mobile cloud storage. *International Journal of Engineering & Technology*, 7(4.36), p.368.