Implementation and evaluation of EMAES – A hybrid encryption algorithm for sharing multimedia files with more security and speed

Original Scientific Paper

Riddhi Somaiya
Saurashtra University, Department of computer science
Rajkot, India
riddhisomaiya@gmail.com

Atul Gonsai
Saurashtra University, Department of Computer Science
Rajkot, India
atul.gosai@gmail.com

Rashmin Tanna
Gujarat technological university, AVPTI, Electronics and Communication department
Rajkot, India
rashminstanna@gmail.com

Abstract *– In this era of smartphones, a huge amount of multimedia files like audio, video, images, animation, and plain text are shared. And with this comes the threat of data being stolen and misused. Most people don't think about the security of data before uploading it to any platform. Most apps used on smartphones upload our data to their server. Not only this, but other third-party apps can also read that data while it is being transmitted. One solution to this problem is encrypting the data before sharing it and decrypting it back at the other end so that even if it is intercepted in between the transmission, it would be impossible to decrypt it. In this paper, a newly designed hybrid encryption algorithm EMAES that includes the efficiency of MAES (Modified Advanced Encryption Standard) and security of ECC (Elliptic Curve Cryptography) was implemented in MATLAB as well as in android studio 4.0. using a mobile messaging application. Also, it was tested for different speeds and security parameters. Further, it was compared with standard algorithms like the RC4, RC6 and Blowfish as well as with other hybrid algorithms like RC4+ECC, RC6+ECC and Blowfish+ECC. The EMAES was found 30% more efficient in terms of encryption and decryption time. The security of EMAES also showed improvement when compared with other hybrid algorithms for parameters like SSIM (structural similarity index measure), SNR (Signal to Noise Ratio), PSNR(Peak Signal to Noise Ratio), MSE (Mean Squared Error) and RMSE (Root Mean Squared Error). And finally, no significant improvement was found in the CPU and RAM usage.*

Keywords*: Cryptography, AES, ECC, EMAES, MAES, RC4, RC6, hybrid encryption algorithm*

1. INTRODUCTION

The most commonly used security encryption algorithm is Rijndael, which is also known as AES (Advanced Encryption Standard) in the standardized form. It is used in the WPA2 security standard for Wi-Fi networking. In our previous research work, we modified the original algorithm and found that its efficiency improved by 68%. The implementation of this Modified AES (MAES) algorithm in MATLAB software was done in [1].

Current research is being done to make MAES more secure; for that, dual-layer security with the combination of another algorithm is proposed. MAES was extended with dual-layer security with the combination of ECC where ECC (Elliptic Curve Cryptography) is used to generate a random key every time for MAES. The hybrid algorithm was then implemented in MATLAB as well as in the Android app for comparison with other standard algorithms like RSA, ECC, AES, RSA+ECC, etc... in terms of efficiency and security. The resultant hybrid algorithm EMAES (ECC and MAES) proved to be more efficient and secure for sharing multimedia files as compared to other algorithms.

The original name of AES (Advanced Encryption Standard) is Rijndael and was selected by NIST during the AES selection process [2]. It is the first and only algorithm that is a publicly accessible cipher approved by the National Security Agency (NSA). It is based on substitution–permutation network design principle and is efficient for both software and hardware. AES performs well on a large variety of hardware, from 8-bit smart cards to high-performance computers. It has a fixed block size of 128 and three categories of key sizes 128, 192, or 256 bits. It operates on a 4 × 4 column array of bytes. The key size used for an AES cipher specifies the number of transformations rounds to produce cipher from plaintext and vice versa, it moves through 4 major functions in each round i.e.

- SubBytes – based on a lookup table each byte is substituted with another by a non-linear substitution step.
- ShiftRows – last three rows of the state are cyclically shifted several times in this transposition step.
- MixColumns – it operates on the columns by combining the 4 bytes in each column of the state by a linear mixing operation.
- AddRoundKey – using bitwise xor each byte of the round key is combined with each byte of the state.

MAES (Modified Advanced Encryption Standard) is the faster version of AES. After reviewing the encryption algorithms, AES was found to be more secure and compatible with both hardware and software [3]. So, we decided to improve its efficiency as per today's requirements. Generating the same sbox and inverse sbox every time was requiring more CPU time. Also, in the mixcolumns part, a large number of multiplication processes were consuming more CPU time.

AES generates an SBOX having 256 entries and an Inverse SBOX by calculating inverse GF (28) of all the 256 entries every time it is initiated. This research eliminates all these calculations by adding a fixed SBOX as well as Inverse SBOX. At the time of the mixcolumns() procedure, AES multiplies all the substituted data which will be one element from SBOX with a poly matrix as shown in Fig. 1.

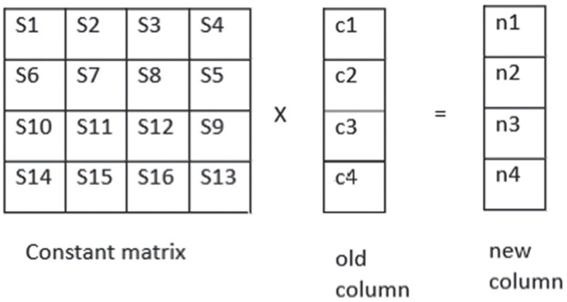


Fig. 1. Mix columns process in AES

To multiply just four bytes from data, the CPU will go through the below-given calculation:

$$SBOX'_{0,c} = (\{02\} \bullet sbox_{0,c}) + (\{03\} \bullet sbox_{1,c}) + sbox_{2,c} + sbox_{3,c}$$

$$SBOX'_{1,c} = sbox_{0,c} + (\{02\} \bullet sbox_{1,c}) + (\{03\} \bullet sbox_{2,c}) + sbox_{3,c}$$

$$SBOX'_{2,c} = sbox_{0,c} + sbox_{1,c} + (\{02\} \bullet sbox_{2,c}) + (\{03\} \bullet sbox_{3,c})$$

$$SBOX'_{3,c} = (\{03\} \bullet sbox_{0,c}) + sbox_{1,c} + sbox_{2,c} + (\{02\} \bullet sbox_{3,c})$$

MAES solves this calculation problem by taking two fixed matrices i.e., SBOX2 and SBOX3. Where in

$$SBOX2_{(R,C)} = SBOX_{(R,C)} \text{ x } 2$$

$$SBOX3_{(R,C)} = SBOX_{(R,C)} \text{ x } 3$$

This will eliminate a lot of excessive computational load on the CPU and increase the speed of operation. Results in [4] show that the percentage improvement in the encryption process is 65.386% as described in Table 1.

Table 1. Comparison of AES and MAES algorithms

Input Data Type	Execution Time		Improvement in efficiency due to Modifications
	AES (second)	Modified AES (second)	
Text (1024 bytes)	7.548	4.506	40.30%
Audio (40000 bytes)	166.633	38.575	76.99%
Image (777845 bytes)	1019.369	215.308	78.87%
Average Percentage improvement in efficiency	65.386%		

ECC (Elliptic Curve Cryptography) algorithm is based on the algebraic structure of elliptic curves over finite fields, public key cryptography is done. Fig. 2 shows examples of such elliptic curves. Elliptic curve-based algorithms use slightly smaller key sizes than the variants of the non-elliptic curve. The disparity in the corresponding key sizes increases significantly with rising key sizes. ECC is a public key cryptography (PKC) that has authentication keys, both public and private over finite fields which are based on elliptic curves [5].

In this paper, the EMAES i.e., ECC + Modified AES is implemented in ANDROID STUDIO 4.0 for encrypting and decrypting data in a Wi-Fi Direct chat application for smartphones. Because it is the only practical way to test the algorithm physically on the network with all its aspects. The application is tested on 5 different android phones having different configurations. Also, EMAES is compared with standard encryption algorithms like Blowfish, RC4, and RC6 in the same scenario as in [6-8]. Similarly, it is also tested and compared with the latest hybrid algorithms. Finally, we could conclude from the results that EMAES is approximately 30% more efficient (speedy), uses 25% fewer resources, and is secure as compared to another standard as well as hybrid algorithms.

1.1. RELATED WORK

An improved hybrid cryptographic framework is presented in [9] for an efficient cancellable biometric authentication system that is more secure against hackers. The main contribution of this work is the incorporation of Rubik's Cube encryption into a hybrid framework containing AES, RC6 and Chaos encryption algorithms.

Experimental simulation results confirm the promising results of the proposed Hybrid Encryption framework for efficiently encrypting stored biometric images. Therefore, it is more suitable for protecting biometric templates compared to traditional methods.

An investigation on secure communications based on hybrid encryption algorithms to improve encryption algorithms for wireless sensor networks was done in [10]. The study proposed an encryption scheme that combines the advantages of AES and ECC. This document uses hybrid encryption technology and selects the AES symmetric encryption algorithm to encrypt the data while the ECC algorithm encrypts the key and the HMAC algorithm to authenticate the message and ensure message integrity. Through simulation verification, it is found that this process significantly improves performance.

The hybrid approach described in [11] combines AES, ECC, and SHA256. Referring to existing methods, the proposed hybrid solution is similar to encrypting both text and images using the AES algorithm. The proposed method is more efficient than the previously considered methods because it is more efficient in encrypting text. The proposed method is less efficient for image encryption than the current method.

Hybrid encryption of cross-border e-commerce information is implemented in [12] through the steps of key and private key generation, key management and distribution, and key exchange in hybrid encryption. Experimental Results Compared to the existing encryption methods, the experimental results show that the hybrid encryption method developed in this paper has a longer decryption time and a reduced data error rate of 2.44 MB, resulting in higher security.

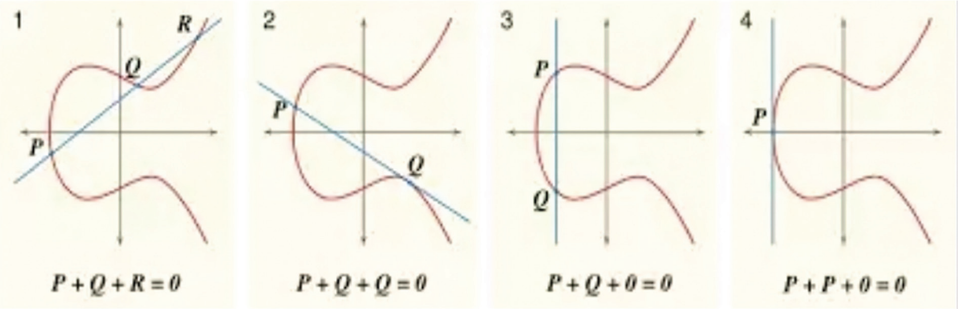


Fig. 2. Examples of Symmetric elliptic curves

In [13] a hybrid encryption method for quantum secure videoconferencing combined with blockchain, and adopt two "one-time pad" and AES quantum encryption methods to solve the problem of the low-key ratio of quantum keys was developed. A cache-efficient query method based on a B+ tree was developed, which was found to be 3.15 times more efficient than the original blockchain query.

As per the authors of [14], the hybrid ECC-AES model was found to take less time than the AES model and other existing models. Current algorithms have certain security issues, such as vulnerability to plaintext attacks, brute-force attacks, side-channel attacks, and computational complexity. The proposed algorithm was able to solve the key exchange problem experienced by AES.

The proposed HAC-based security authentication method [15] achieves a minimum communication cost of 0.017 seconds, a calculation time of 0.060 seconds, and minimum memory usage of 2.502MB, respectively. Hybrid cryptography functions in two ways. One relies on Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC), and the other on Rivest Shamir Adleman (RSA) and AES.

In [16] proposed the idea to use Blowfish for encryption, Message Digest 5 (MD5) for integrity, and Elliptic Curve Diffie Hellman (ECDH) for authentication. The proposed algorithm gives the best results when using two computers (A and B) compared to many other algorithms in terms of ciphertext size, encryption time, decryption time, and throughput.

AES, ECC, and Serpent were used to design an encryption scheme to secure data in an IOT-enabled system. The proposed scheme [17] improves security measures using both symmetric and asymmetric cryptosystems. A two-dimensional classification of existing studies on hybrid cryptography models based on processing phase and scope is presented in [18]. As a result, we can compare the study with other current models that help improve the performance of hybrid models after this COVID-19 pandemic.

Article [19] presents a general model of various hybrid encryption schemes that improve data security. This white paper also presents a comparative study of various traditional and hybrid models actively used for data security. The hybrid scheme can provide a higher level of security than AES and should be chosen if maximum security is required.

Security issues of information transmission and methods of hybrid encryption algorithms were described in [20]. It also considers and analyzes the different characteristics of algorithms on different systems and some common cases of hybrid cryptography, demonstrating the advantages of combining them. A hybrid encryption algorithm enhances transmission security without causing additional problems. It also explains how, for example, cryptographic algorithms can be combined to increase security.

Lightweight hybrid cryptography techniques were explored in [21], primarily using a set of rules based entirely on AES for plaintext encryption and the Elliptic Curve Diffie-Hellman (ECDH) protocol for key encryption. The simplicity of the AES implementation makes it easy, and the complexity of ECDH makes it secure. The design is simulated in Spyder Tool, and Modelsim and implemented in Xilinx Vivado. The effect shows that the proposed lightweight model offers a normal level of security with reduced computational power. Along with the realization of a project to implement multimedia input on his FPGA, a key authentication system for enhanced security was proposed.

2. IMPLEMENTATION ENVIRONMENT

2.1.1. ANDROID

Android is a mobile operating system designed for smart devices such as smartphones, smart tv, smartwatch, etc. It is developed by Open Handset Alliance and commercially sponsored by Google. It is an open-source and free [22] operating system. It has been the best-selling operating system in the world since 2011. As of March 2020, the app store i.e., Google play store features more than 2.9 million applications.

2.1.2. ANDROID STUDIO

Designed specifically for Android development and built on JetBrains' IntelliJ IDEA software, Android Studio is the Android operating system's official IDE (Integrated development environment). It supports developers to design, code, test and launch the application easily and fast. It contains various tools for learning android applications, designing the user interface, coding, compiling and debugging environments along with various testing features.

Developers can create virtual android devices to test the application. Android Studio also supports the installation of the application on real android devices and the logging of performance statistics.

2.1.3. ANDROID APPLICATION

A software application developed to run on android supported devices. It is distributed as a .apk file that contains all the resources of that application. Android apps could be coded in various languages such as java, c++, kotlin, etc., using an android software development kit and JVM i.e., Java Machine. The official development environment is called Android Studio.

2.1.4. WIFI DIRECT

Wi-Fi CERTIFIED Wi-Fi Direct® is a Wi-Fi connection without the requirement of a wireless router or an internet connection. Like Bluetooth, Wi-Fi is a way of communicating wirelessly. The concept of "ad-hoc" Wi-Fi mode has similarities with Wi-Fi Direct [23]. However, Wi-Fi Direct has an easier way to automatically discover and connect to nearby devices like cameras, Mobile phones, PCs, printers and gaming devices compared to an ad-hoc Wi-Fi connection. Using the latest Wi-Fi security i.e., Wi-Fi Protected Setup™ supported devices, one can make a point-to-point connection or a group of several devices can connect simultaneously and exchange files, play media, print documents or display screens between them [24].

3. PROPOSED WORK

3.1.1. EMAES

The EMAES is an improved version of MAES where MAES improves the efficiency of AES and EMAES provides better security to it. AES is almost impossible to crack without the knowledge of its KEY. As we used MAES in a mobile chat application, it was necessary to share the key with the receiver so that the data could be decrypted. When we share the key wirelessly, it becomes vulnerable to attacks from third-party intruders. i.e., hackers/crackers who can read the key and then decrypt the data easily. As a solution to this, we used the ECC algorithm that helped to generate random private and public keys. Here, both devices (same application in two different smartphones) share their public key and create a shared key with the help of their private key and others' public key. Now this shared key is used as KEY to encrypt and decrypt the data in MAES.

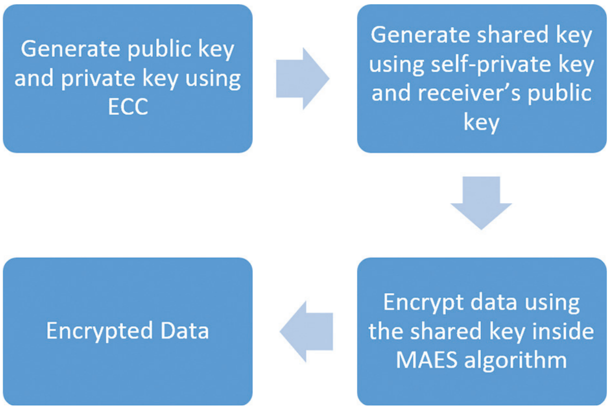

Generate public key and private key using ECC
Generate shared key using self-private key and receiver's public key
Encrypt data using the shared key inside MAES algorithm
Encrypted Data

Fig. 3. EMAES algorithm Encryption process

The EMAES algorithm encryption process is described in Fig. 3. At first, 256 bits of public key and private key will be generated. The sender and the receiver will have to exchange their public keys to share the data. A shared key will be generated using its own private key and the receiver`s public key. This shared key will be used as the KEY in the MAES algorithm to encrypt the data.

The decryption process of the EMAES algorithm will be the same as the Encryption process. As shown in

Fig. 4, the decryptor end will generate a public key and a private key. Also, a shared key using a self-private key and the sender's public key will be generated. Then the shared key will be used in the MAES algorithm as a secure key and the encrypted data will get decrypted.

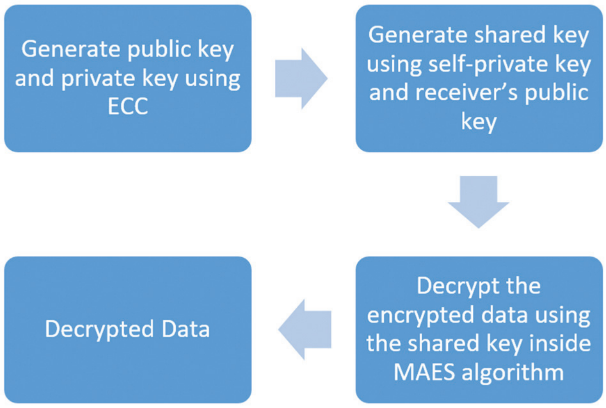

Generate public key and private key using ECC
Generate shared key using self-private key and receiver's public key
Decrypt the encrypted data using the shared key inside MAES algorithm
Decrypted Data

Fig. 4. EMAES algorithm decryption process

Here, the shared key in ECC can be generated in any size i.e., 128, 198, or 256 bits according to the requirement of MAES.

This way EMAES algorithm gives us all the advantages of AES along with improved speed through MAES and better security through ECC. We get dual layer security where data and key both are hidden from third parties.

3.1.2. EMAES chat app

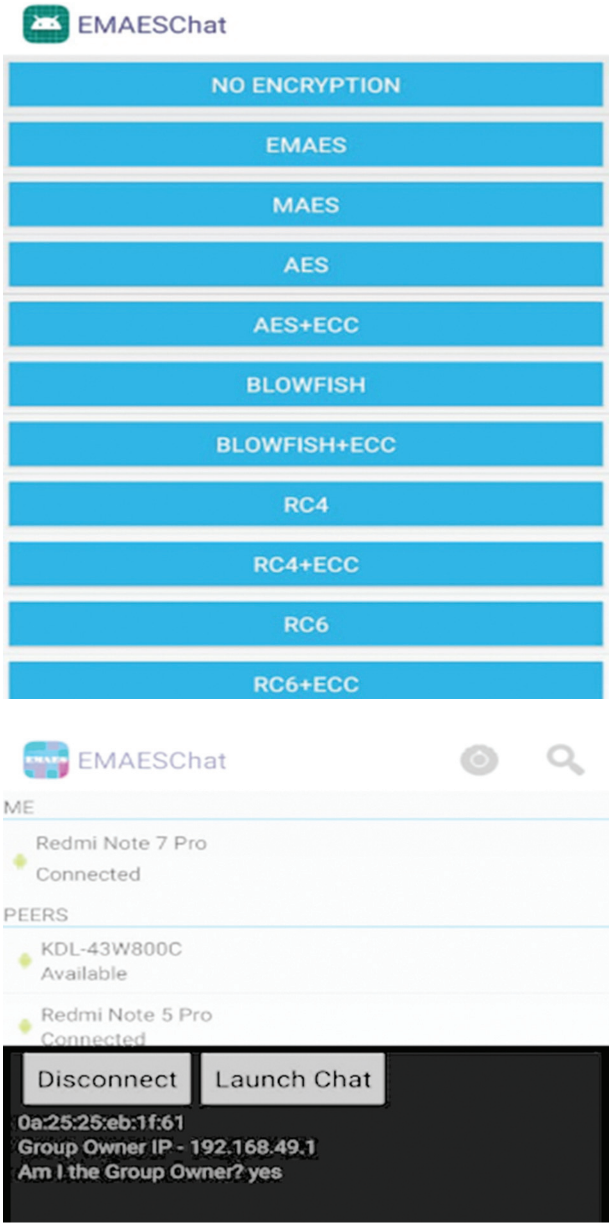
EMAES chat app is a testing tool in the form of a chat application on the android platform. One can select any encryption algorithm provided in it and test them all one by one. This application requires Wi-Fi and location service to be kept ON. The user can connect to another phone and then test the selected encryption algorithm by sending and receiving multimedia messages. The messages sent and received through the application are encrypted as well as decrypted using the selected encryption algorithm. The test parameters recorded in the google firebase real-time database are later used for the comparison of algorithms.

3.1.3. Application Flow

Fig. 5 shows the connection screen of the app encryption algorithm for testing purposes. Currently available algorithms are EMAES (proposed in this article), MAES, AES, AES+ECC, BLOWFISH, BLOWFISH+ECC, RC4, RC4+ECC, RC6, RC6+ECC. The option of NO ENCRYPTION is also available which sends data without encryption. On the next screen, the user can connect with another android phone with the EMAESChat app installed and opened.

3.1.4. Chat room screen

After starting the chat room, the application instantly generates the private and public keys and sends them to the other user only if one of the algorithms with ECC is selected. Similarly, the app receives the public key of the opposite user and generates the shared key. In other cases, a simple 32-bit constant key is selected as the public key. Users can send text messages as well as multimedia files like drawings, images, audio, video, and other file formats on the chatroom screen. A screenshot of the chat room screen is shown in Fig. 6. When the user presses the send button, the application converts the data selected to be sent in string format and encrypts that string with the selected encryption algorithm. At the receiver end, the application receives the encrypted message and decrypts it with the selected algorithm using a generated shared secret key. Its decryption results in a string that is converted into the exact original message sent by the sender. The same process is repeated every time the user sends and receives the messages. The shared key will expire when the user exits the chat room if it is previously generated by the system.



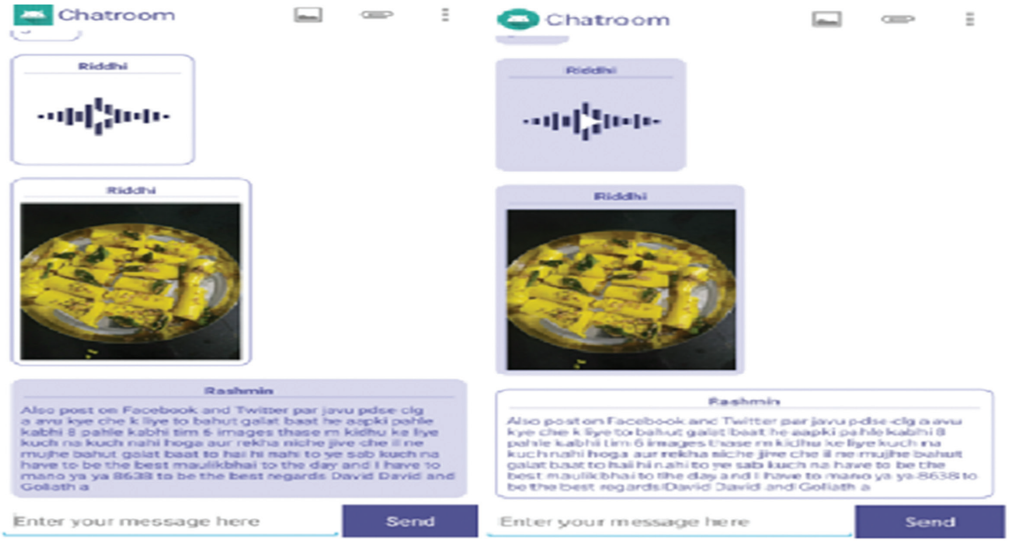
Redmi Note 7 Pro
Connected

KDL-43W800C
Available

Redmi Note 5 Pro
Connected

0a:25:25:eb:1f:61
Group Owner IP - 192.168.49.1
Am I the Group Owner? yes

Fig. 5. Screenshot of connection screen



Also post on Facebook and Twitter par javu pdse clg a avu kye che k liye to bahut galat baat he aapki pahle kabhi 8 pahle kabhi tim 6 images thase m kidhu ke liye kuch na kuch nahi hoga aur rekha niche jive che ll ne mujhe bahut galat baat to hai hi nahi to ye sab kuch na have to be the best maulikbhai to the day and I have to mano ya ya 8638 to be the best regards David David and Goliath a

Also post on Facebook and Twitter par javu p-dse-clg a avu kye che k liye to bahut galat baat he aapki pahle kabhi 8 pahle kabhi tim 6 images thase m kidhu ke liye kuch na kuch nahi hoga aur rekha niche jive che ll ne mujhe bahut galat baat to hai hi nahi to ye sab kuch na have to be the best maulikbhai to the day and I have to mano ya ya-8638 to be the best regards!David David and Goliath a

Fig. 6. Screenshot of chatroom screen sending and receiving messages

4. RESULTS

In Table 2 multimedia files like text, image, audio, and video files are used to encrypt and decrypt with EMAES. This table shows the size of each file and the time taken to encrypt and decrypt the data. The original file is compared with the decrypted file to search for errors or noise using the parameters like SSIM (structural similarity index measure), SNR (Signal to Noise Ratio), PSNR(Peak Signal to Noise Ratio), MSE (Mean Squared Error) and RMSE (Root Mean Squared Error). The data after decryption was found exactly the same as the original. Also, table 2 shows the utilization of resources like CPU, RAM, and NETWORK while encrypting, sending, receiving and decrypting the data, similar to the work done in [25]. Here, 56kb of text took 0.049 seconds to encrypt and 0.035 seconds to decrypt with 0 noise and errors. Same way, 1.82 MB of audio took 1.1 seconds to encrypt and 0.52 seconds to decrypt. Many other file types like .wav, .pdf, .doc, etc., were also tested successfully during the research.

Table 2. Proposed work implemented on multi-Media files

PARAMETERS	TEXT	AUDIO	IMAGE	VIDEO
Size	52 kb	1.82MB	4MB	27MB
Encryption Time (sec)	0.049	1.1	36.40	93.16
Decryption Time (sec)	0.035	0.52	17.99	54.21
SSIM	1	1	1	1
PSNR	INF	INF	INF	INF
SNR	0	0	0	0
MSE	0	0	0	0
RMSE	0	0	0	0
CPU (%)	12	24	29	37
RAM (MB)	121	131	261	315
NETWORK (bps)	39.23	49.90	59.10	65.32

Table 3 depicts the comparison of EMAES with standard algorithms like Blowfish, RC4, and RC6. All the algorithms were implemented in the EMAESChat app and compared based on various parameters while sending and receiving messages. 10kb, 100kb, and 1 MB data were considered for small, medium, and large sizes respectively. The parameters considered for comparison were Encryption time, Decryption time, SSIM, SNR, and MSE. The table also shows the utilization of resources like CPU, RAM, and NETWORK while sending and receiving multimedia files. A comparison of EMAES with hybrid algorithms like Blowfish+ECC, RC4+ECC, and RC6+ECC is depicted in table 4. The comparison parameters used were the same as that in table 3.

A comparison of EMAES with both standard and hybrid algorithms shows that EMAES provides better security as it is completely based on AES. Also, it has the advantages of ECC. Moreover, its strength can be seen by comparing SSIM (Structure Similarity Index), SNR (Signal to Noise Ratio), and MSE(Mean Squared Error) values, as they are more reliable than that of any other algorithms.

A chart based on table 3 is shown in Fig. 7. This will allow us to visualize the numerical differences between EMAES and other standard algorithms with the parameters considered for testing. It could be seen from the chart that the average execution time (encryption and decryption) of EMAES is 0.90 sec which is the fastest. The second fastest is Blowfish with 1.29 sec. Hence, we can say that EMAES is at least 30% faster than all the standard algorithms. Also, the resource utilization is 25% less than other algorithms. But for this security is not compromised.

Fig. 8. represents a chart based on Table 4. With the help of this, comparison parameter values used to compare the EMAES with other hybrid algorithms could be analyzed. It was seen that the average execution time of EMAES was 0.90 sec and that of the fastest hybrid algorithm i.e., Blowfish+ECC was 0.98 sec which is still 8.1% faster. Also, resource utilization and security were not compromised.

Table 3. Comparison of EMAES with standard algorithms

Comparison Parameters/ Algorithms	Data size (Small / Medium / Large)	Encryption Time (seconds)	Decryption Time (seconds)	SSIM (Structured Similarity Index)	SNR (Signal-to-Noise Ratio)	MSE (Mean Squared Error)	CPU usage (%)	RAM usage (kb)	Network throughput (bps)
EMAES	S	0.05	0.5	0.9	0	0	12	121	39.23
	M	0.51	0.74	0.9	0	11	24	131	49.9
	L	1.78	1.86	0.9	-1.3	5.3	29	261	59.65
Blowfish	S	0.07	0.07	1	0	0	12	122	10.1
	M	0.95	0.71	0.96	65	78	34	101	35.6
	L	1.13	1.1	0.99	-1.8	4.2	21	168	65.3
RC4	S	0.03	0.04	1	0	0	18	141	23.1
	M	0.31	0.38	0.96	0	70	22	137	61.1
	L	3.32	2.7	0.99	0	2.4	29	161	78.8
RC6	S	0.59	0.91	0.99	0	2.2	16	204	20.6
	M	0.58	0.59	0.9	0	121	19	246	22.3
	L	2.64	1.95	0.99	-2.2	4.8	21	303	32.8

Table 4. Comparison of EMAES with hybrid algorithms

Comparison Parameters / Algorithms	Data size (Small / Medium / Large)	Encryptio Time (seconds)	Decryption Time (seconds)	SSIM (Structured Similarity Index)	SNR (Signal-to-Noise Ratio)	MSE (Mean Squared Error)	CPU usage (%)	RAM usage (kb)	Network throughput (bps)
EMAES	S	0.05	0.5	0.9	0	0	12	121	39.23
	M	0.51	0.74	0.9	0	11	24	131	49.9
	L	1.78	1.86	0.9	-1.3	5.3	29	261	59.65
Blowfish+ECC	S	0.45	0.07	1	0	0	12	122	10.9
	M	0.75	0.71	0.96	65	78	34	123	36.1
	L	1.93	1.97	0.99	-1.8	4.2	21	231	67.3
RC4+ECC	S	0.04	0.04	1	0	0	18	146	26.1
	M	0.32	0.38	0.96	0	70	22	154	61.9
	L	3.21	3.1	0.99	0	2.4	29	211	79.4
RC6+ECC	S	0.6	0.98	0.99	0	2.2	16	211	23.2
	M	0.72	0.66	0.9	0	121	19	255	25.3
	L	2.89	2.1	0.99	-2.2	4.8	21	335	34.8

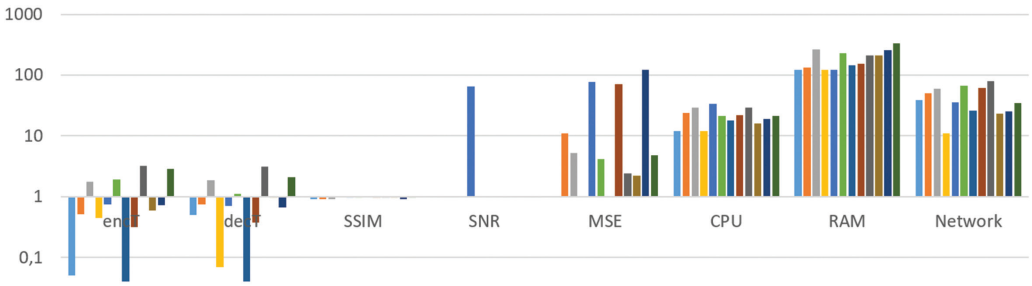


Fig. 7. Visual representation of comparison parameters of EMAES and other standard algorithms in logarithmic scale to the base 10

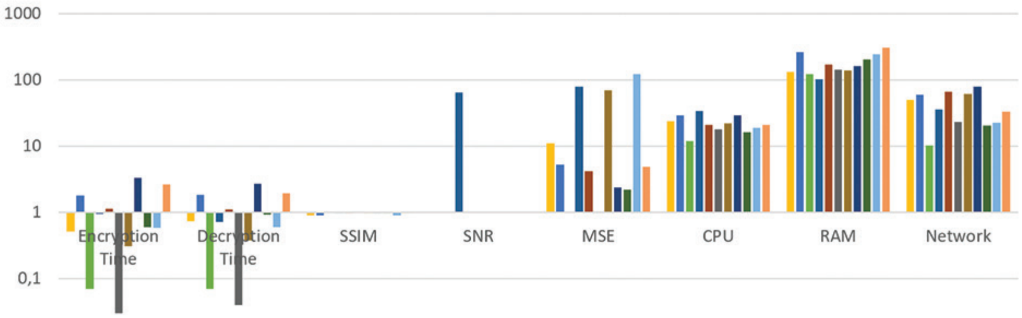


Fig. 8. Visual representation of comparison parameters of EMAES and other hybrid algorithms in logarithmic scale to the base 10

5. CONCLUSION

This research proposes a new hybrid algorithm that gives multiple benefits of speed, accuracy, and security with minimum resource utilization.

The algorithm was previously designed as MAES i.e., modified AES to increase efficiency. But AES is secure enough only until its key is hidden. When we need to share the data with another device, we also need to share the key to decrypt the data. This was a big risk to data security. Then the combination of MAES with ECC improved the security by making it a dual layer. Here, AES is well known for data security while ECC gives a strong public key technique that is next to impossible to hack.

The algorithm was previously tested in MATLAB to check its security and efficiency against other encryption algorithms, the positive outcomes encouraged us to test it in a live environment. Hence the EMAES was implemented in an android chat application i.e., the EMAESChat app which works on Wi-Fi-direct and was tested in multiple android smartphones sharing different kinds of multimedia files.

Finally, by comparing the results with standard as well as hybrid algorithms, it could be concluded that EMAES is on average at least 30% faster than the fastest algorithm i.e., blowfish, and 8% faster in execution time when compared with the fastest hybrid algorithm i.e., blowfish+ECC. In terms of resource utilization like CPU, RAM, and network also EMAES is at least 25% better. The most significant feature of EMAES is that it does not compromise security while achieving efficiency.

Future work could include the implementation of EMAES on FPGA and test with a large number of devices on the Internet. It could also be tested in cloud computing and IoT environments. There is also the scope for comparing it with other live streaming and video calling algorithms.

6. REFERENCES

[1] R. Tanna, R. Tanna, J. Bhadeshiya, "Improvement in the execution time of AES algorithm by modifications in sbox and mix columns for multimedia applications", International Journal of Research in Engineering, IT and Social Sciences, Vol. 8, No. 4, 2018, pp. 65-68.

[2] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, E. Roback, "Report on the Development of the Advanced Encryption Standard (AES)", Journal of research of the National Institute of Standards and Technology, Vol. 106, No. 3, 2001, pp. 511–577.

[3] R. C. Somaiya, A. M. Gonsai, R. S. Tanna, "WLAN security and efficiency issues based on encryption techniques", International Journal of Research in Engineering, IT and Social Sciences, Vol. 6, No. 9, 2016, pp. 27-32.

[4] R. Somaiya, A. Gonsai, R. Tanna, "Design and Implementation of a New Encryption Algorithm in MATLAB for Multimedia Files", Vidhyayana, Vol. 6, No. 6, 2021, pp. 985-999.

[5] X. Li, J. Chen, D. Qin, W. Wan, "Research and realization based on hybrid encryption algorithm of improved AES and ECC", Proceedings of the International Conference on Audio, Language and Image Processing, Shanghai, China, 23-25 November 2010, pp. 396-400.

[6] H. K. Verma, R. K. Singh, "Enhancement of RC6 block cipher algorithm and comparison with RC5 & RC6," Proceedings of 3rd IEEE International Advance Computing Conference, Ghaziabad, India, 22-23 Feb. 2013, pp. 556-561.

[7] B. Schneier, "Description of a new variable-length key, 64-bit block cipher (Blowfish)", International workshop on fast software encryption, Springer, Vol. 809, No. 6, 1994, pp. 191-204.

[8] R. Singh, "An overview of android operating system and its security features", International Journal of Engineering Research and Applications, Vol. 4, No. 2, 2014, pp. 519-521.

[9] M. Helmy et al. "A hybrid encryption framework based on Rubik's cube for cancelable biometric cyber security applications", Optik, Vol. 258, No. 168773, 2022, pp. 30-42.

[10] Y. Huiwei, "Application of hybrid encryption algorithm in hardware encryption interface card", Security and communication networks, Vol. 2022, No. 7794209, 2022, pp. 1-11.

[11] P. William, A. Choubey, G. S. Chhabra, R. Bhattacharya, K. Vengatesan, S. Choubey, "Assessment of Hybrid Cryptographic Algorithm for Secure Sharing of Textual and Pictorial Content", Proceedings of the International Conference on Electronics and Renewable Systems, Tuticorin, India, 16-18 March 2022, pp. 918-922.

[12] J. Li, S. Wang, Z. Zhang, Y. Xu, "Research on Hybrid Encryption of Cross Border E-commerce Transaction Information Based on B+ Search Tree Algo-

rithm", Proceedings of the 2nd EAI International Conference on IoT and Big Data Technologies for Health Care, Leicester, UK, 18-19 October 2021, pp. 290-307.

[13] D. Zhu, J. Zheng, H. Zhou, J. Wu, N. Li, L. Song, "A Hybrid Encryption Scheme for Quantum Secure Video Conferencing Combined with Blockchain", Mathematics, Vol. 10, No. 17, 2022, pp. 30-37.

[14] S. Ahmad, S. Mehfuz, J. Beg, "Hybrid cryptographic approach to enhance the mode of key management system in cloud environment", Journal of Supercomputing, Vol. 78, No. 17, 2022, pp.1-37.

[15] S. P. Kavitha, I. Mandal, C. Rangaswamy, "Hybrid and Adaptive Cryptographic-based secure authentication approach in IoT based applications using hybrid encryption", Pervasive and Mobile Computing, Vol. 82, No. 101552, 2022, pp. 8-25.

[16] H. A. Abdulhameed, A. A. Abdulhameed, M. F. Mosleh, A. T. Mohammad, "Lightweight security protocol for WSNs using hybrid cryptography algorithm", AIP Conference Proceedings, Vol. 2547, No. 1, 2022, p. 060006.

[17] S. Das, S. Namasudra, "A Novel Hybrid Encryption Method to Secure Healthcare Data in IoT-Enabled Healthcare Infrastructure", Computers & Electrical Engineering, Vol. 101, No. 107991, 2022, pp 107-115.

[18] A. A.-R. El-Douh, S. F. Lu, A. Elkony, A. S. Amein., "A Systematic Literature Review: The Taxonomy of Hybrid Cryptography Models", Lecture Notes in Networks and Systems, Springer, Vol. 439, 2022, pp. 714-21.

[19] P. Soni, R. Malik., "A Comparative Study of Various Traditional and Hybrid Cryptography Algorithm Models for Data Security", Modeling, Simulation, and Optimization, Springer Nature, Vol. 292, No.28, 2022, pp. 31-47.

[20] Q. Zhang, "An Overview and Analysis of Hybrid Encryption: The Combination of Symmetric Encryption and Asymmetric Encryption", Proceedings of the 2nd International Conference on Computing and Data Science, Stanford, CA, USA, 28-29 January 2021, pp. 616-622.

[21] P. Kitsos, G. Kostopoulos, N. Sklavos O. Koufopavlou, "Hardware implementation of the RC4 stream cipher", Proceedings of the 46th Midwest Symposium on Circuits and Systems, Cairo, Egypt, 27-30 Dec. 2003, pp. 1363-1366.

[22] D. Camps-Mur, A. Garcia-Saavedra, P. Serrano, "Device-to-device communications with Wi-Fi Direct: overview and experimentation", IEEE Wireless Communications, Vol. 20, No. 3, 2013, pp. 96-104.

[23] T. Alam, M. Aljohani, "An approach to secure communication in mobile ad-hoc networks of Android devices", Proceedings of International Conference on Intelligent Informatics and Biomedical Sciences, Okinawa, Japan, 28-30 November 2015, pp. 371-375.

[24] S. K. Ghosh, S. Rana, A. Pansari, J. Hazra, S. Biswas, "Hybrid Cryptography Algorithm for Secure and Low-Cost Communication", Proceedings of International Conference on Computer Science, Engineering and Applications, Gunupur, India, 13-14 March 2020, pp. 1-5.

[25] A. Mammenp, S. KN, R. Bhakthavatchalu, "Implementation of Efficient Hybrid Encryption Technique", Proceedings of the 2nd International Conference on Intelligent Technologies, Hubli, India, 24-26 June 2022, pp. 1-4.