Computers and Electrical Engineering 104 (2022) 108467

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

A novel symmetric cryptographic method to design block complexity for data security

B. Umapathy, G. Kalpana [*]

Department of Computer Science, College of Science and Humanities, SRM Institute of Science and Technology, Chennai, India

A R T I C L E I N F O

Editor: Dr. M. Manikandan

Keywords:
Cryptography
Cloud storage
Encryption
Cloud computing

A B S T R A C T

The growth and development of Cloud Computing from its inception till today has made a tremendous progress and the evident of the same is seen in every walk of life. However considering data security there is a big hiatus. It is consistently under threat besides many guards**.** This in turn has created a research gap to find out the secure guard for Data Security. Data security is primarily dependent on encryption algorithms to secure data in the CC environment. Thus the encryption algorithm plays a predominant role in Cloud Storage (CS) which is a place to store all the data of the users or data owners. Hence the usage of CS has become inevitable in almost all fields including education, industry, business and the usage of the same is grown in leaps and bounds particularly in the pandemic. The metamorphosis of the encryption algorithm (i.e. AES, Triple DES, and Blowfish) has a complex formation to secure the data. However complex the formation is; it fails to cope with the modern-day processer evaluation. The complexity level tends to be low considering the performance of the modern-day processer evaluation. Hence an alternate solution is demanded and this research paper has attempted to meet the demand by introducing the new multi-layered cryptographic algorithm named as UK to secure the CS data. This UK algorithm follows an entirely new transformation procedure to secure the data. It creates huge complexity in reasonable encryption time, even though brute force needs to be more complex ever. The key and Metadata are not stored in any place so that the CS service provider will not be able to do anything in this structure. In this case, the database service providers have the hold of the encrypted data of the user which cannot be accessed by any attacking techniques. Hence, the CS provider becomes the warehouse maintainer.

1. Introduction

Cloud computing environment is secured by using cryptographic algorithm. Cryptographic is an art that converts the plaintext to cipher text using mathematical operation, logical operation, permutation and combination. In spite of the multi-layered operations of the Cryptographic algorithm the intruders get a chance to easily intervene at the single level encryption algorithm. To avoid this intrusion multilevel encryption algorithm like Triple DES [1] is introduced. Thus a primary attempt is made to protect the cloud data security. The essential initiative of the Cloud data security is Availability, Confidentiality, Accountability, and Integrity. Pertaining to this credibility the CS facility is in high demand and much sought after. This has paved a way to the advance development and hence refurbishment of cryptographic has become a must to meet the requirements of the Morden-days [2].

* Corresponding author.
E-mail address: kalpanag@srmist.edu.in (G. Kalpana).

https://doi.org/10.1016/j.compeleceng.2022.108467
Received 25 May 2022; Received in revised form 28 October 2022; Accepted 2 November 2022
Available online 18 November 2022

Perpetually the data security domain has got even more progressed in late 1990s under the National Institute of Standards and Technology (NIST). The DES, AES, RSA cryptographic algorithms also got developed simultaneously. In 2001 NIST approved the AES (Rijndael) as a reliant secure for holding documents in encrypted form. Owing to its excellent performance, the AES has paved a gate way later to the U.S Government, defence and FBI to use it across platforms, good RAM and ROM management, faster key setup, and key expansion technique and block cipher method. From 2001 to till now AES algorithm is being employed in all industries and it has evolved to be one of the leading cryptographic algorithms ever [3].

For both encryption and decryption symmetric cryptographic algorithm secret key is used alike. While there are two divisions in symmetric encryption namely; Stream Ciphers and Block Ciphers this paper is limited to focus only on the Block Ciphers to build the novel UK algorithm. The paper is divided into five sections and each section discusses the research findings accordingly. Section 1 deals with the introduction of the research work. Section 2 focuses on the related work of the research. Section 3 concentrates on the proposed UK algorithm. Section 4 substantiates the results and discussion and Section 5 concludes the summarisation of the research and throws light on the further research scope in future.

1.2. Motivation

The cloud storage service provider like Google drive, Drop box, One drive, and many alike are still using the AES encryption algorithm to secure the user's data in cloud environment. The master key's used for the encryption and decryption is automated with user account details and Meta-data also maintains the cloud service provider. At this avenue user's details may have a chance to lose its security by server hacking or untrusted admin and thus in turn the master key could be tampered. If it happens so, the primary function of the Metadata to maintain the security will be failed and the Meta data itself will leak the basic information about the encrypted data hold by the users. It is then the novel cryptographic algorithm comes in rescue. The motive of novel cryptographic algorithm is to create a complex transformation method from plaintext to cipher text, with high complexity in average encryption and decryption time, and create multi-layered security system by employing multiple passwords which is given directly by the user. The metadata should not be maintained in the cryptographic algorithm. It should be compatible for image, video, audio, document and text.

2. Related work

AES has varied categories based on the secret key size (no of bits). AES has three categories AES-128, AES-192 and AES-256. The no of encryption round is also increased based on the key size. The data block size is fixed as128 bits for all three categories. The AES-256 is a high efficient and complex algorithm which holds 14 rounds. The round sub-keys is derived from the master secret key based on S-box [4]. Kumar and Karthigaikumar [5] have proposed a novel key expansion for AES algorithm that would not compromise the security and it would as well improve the execution speed. The CFA-based pipeline architecture achieves the improvement of about 2.53% in throughput.

The light-weight efficient cryptographic algorithm in generation process as postulated by R Bharathi and N. Parvatham [6], is very useful to employ in the IOT using FPGA prototyping. Their work involved a very confusing combination by increasing the number of rounds as much as 16, 32, 64, through parallel operation. This helped to improve the Latency which even takes lesser time to execute.

Tahir et al. [7], have proposed a new method that relies on CryptoGA to handle privacy issues, data integrity, and many alike. GA is used to generate decryption and encryption keys that are integrated with crypto systems to ensure the integrity and privacy of cloud information. The comparative evaluation considers common and known variables: throughput, execution time, avalanche effect, and key size. Ten different databases are used for research purposes for validation and testing purposes. Test results show that the method presented guaranteed integrity and protects the privacy of user information from third parties.

Ferdush et al. [8], suggested a chaos-based lightweight image encryption technology. First, they proposed two chaotic maps, a common algorithm and framework based on Logistic Arnold for optical image encryption, and researched further. They examined various image clusters such as textures, others, medical, underwater, and many more. Their result showed that Light weight image encryption technology offers a potential way to reduce computational complexity. Park et al. [9], proposed an effective parallel implementation of the Simeck family block cipher on the new 64-bit Intel processor. It was very efficient when it is used for the adopted encryption scheme to load balance (LB) the resulting big data.

DES key generation continues the 64-bit key by discarding every 8 bits of the original key, converting it to a 56-bit key, and splitting it into two 28-bit parts. The DES operation is through bit shift, byte substitution [10]. Key generation for the Blowfish algorithm varies between 32 and 448 bits. This flexibility adds to the confusion of decryption using XOR [11]. The strength of the encryption and decryption algorithms depends on the difficulty of key detection. In the above investigation, the key size is mostly fixed for most cryptographic algorithms, with the exception of the Blowfish algorithm, which has a limit of up to 448 bits. Key generation is a very important part of the ransom ware infection model, and threat solutions are easily protected by a defence-in-depth approach [12]. Symmetric and asymmetric algorithms such as AES, DES, RSA, and BLOWFISH are tested in different environments and in different file types (image files, binary files, text files, and many more.). Of the all, AES turned out to be the most powerful algorithm [13].

Song et al. [14] have developed a new ABE method that splits the key extraction and attributes checking functions so that KGC does not recognize a particular attribute of the user and AAC does not get the user's key. The user has presented a dummy token to KGC to get each dummy key. Here the user can extract the final private key. Mudepalli et al. [15] have proposed highly secure data by uploading technique using elliptic curve key for key generation and data encryption is done by blowfish algorithm. This technique is highly efficient for both the data user and data the owner. Haq et al. [16] have upgraded the blowfish algorithm for the digital image to increase mathematical complexity by using substitution boxes.

Barman et al. [17] have proposed a new concept where the two communicating users who would share a session key. The cryptographic framework is built utilizing their biometric information. Two communicating parties' biometric information is merged to create a mutual locker in a cryptographic framework. A mutual locker secures biometric data against interception during locker distribution over a public network and allows users to lock their biometric data with each other. A mutual locker also shields the counterparty's biometric information from an insider attacker who uses their personal key to communicate with them legitimately. Krishnaveni et al. [18] proposed an intrusion detection system with a combination of selected features to overcome honeypot attacks and improved results based on the Kyoto dataset.

Das et al. [19] introduced an open standard to take standardization and serve as a guide for laying secure modules into practise. Trusted Third Parties are switched in real-time to ensure low dependency. Decentralization of secret storage through the use of a dependable third party that can be quickly replaced by other vendors who follow the criteria is made possible by dividing the secret information used to create secret keys into separate, independent parts. The standard makes sure that the secret keys will only have temporary validity during the procedure, rendering a key that has already been used will turn useless to a potential attacker. This will be made possible by allowing users to select the key generation and authentication solutions they want, giving them control over their own secrecy.

Atiewi et al. [20], proposed IOT-based multi-factor authentication and lightweight cryptographic schemes in cloud storage environments. IOT devices are organized in Sensitive and non-confidential data as follows: Confidential data is divided into two parts. Encrypted by a separate encryption algorithm (RC6, Feistel) and data storage in a private cloud Memory to ensure high security. The less sensitive data is encrypted by a single algorithm (AES). It is stored in the public cloud. Multi-factor certification is guaranteed by a trusted authority and User IP, password, biometrics as user identification is enabled.

3. Proposed work

The present research paper focuses on the novel symmetric block cipher UK algorithm to increase the complexity level of the block, to reduce the number of round substitutes using the derived key by the key expansion algorithm. The password is employed only by the user, and maintaining the Meta-data is avoided. The step by step procedure of UK algorithm for key generation, encryption, and decryption procedures are follows.

3.1. Key generation process

1. Get five keys from the user.
2. The first three keys shall be in any format and length. Then it enters into the key builder function which holds a unique way to convert the keys into binary digits.
3. Once the keys are converted, they will be unique in size.
4. The fourth key can only be a numeric value.
5. The fifth key will be a file format.
6. Generate keys are created using the combination of the first three keys and the fourth key.
7. Generate_key1 » is the multiplication of the first key and fourth key.
8. Generate_key2 » is the multiplication of the second key and fourth key.
9. Generate_key3 » is the multiplication of the third key and fourth key.
10. After all the above processes are completed, the first three keys are expanded to fixed key length (359,950).
11. There are two cases, Named case1 and case 2, which create confusion and combination.
12. The cases lies in 6 combination ((1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)).
13. Case1 is defined by the sum of the total length of the first three keys.
14. Case2 is defined by the sum of generate keys.

In the key generation process, the user needs to give five keys for the encryption. The second and third could be the optional key but the first, fourth and fifth key is a must. While comparing to another encryption algorithm number of the key requirement is more but to achieve the most complex form in a reasonable time it is mandatory. The first three keys can be of any length and format but, a minimum of one binary value is mandatory. The novel key generation process holds the unique way to convert the characters to binary form, if the user uses more binary digits in the password it results in the invariable length of the binary password and it will be more complex to predict the formation of the encryption key. The strength of the key depends upon the number of binary digits used in the password with good intervals. The fourth key must be numerical. The fifth key is used only in the decryption process (file type), which would help to avoid maintaining the Meta-data of the file. The optional keys consider the strength level of the encryption keys. The number of encryption rounds will be the same irrespective of optional keys.

The key generation algorithm generates three values using the multiplication of the length of the first three keys and fourth key individually to determine the number of repetitions. After the process, the invariable key is expanded to the fixed key size as 359,950 bits. Now the complex three encryption keys are ready. To create more confusion and combination by defining the six case combinations is listed in the above procedure [3.1]. Using these case combination two cases are declared. The first case combination is derived by the sum of the invariable length of the encryption keys. The second case combination is derived by the sum of generated keys. Finally, the key generation process results in three encryption keys, three generated values for repetition, Case 1 and 2 starting combinations.

3.2. Encryption process

1. Convert the original file into plaintext (binary format), the size of the plaintext depends upon the file size of the original file.
2. Divide the plaintext into chunks. The size of the chunk is fixed (360,000 bits).
3. No chunks are depended upon the plaintext size.
4. Divided each chunk will undergo three-layered encryption.
5. Each layer of the encryption process uses the respective key and generate key, according to the case's combination.
6. Each layer's first stage is to perform the right shift operation. The shift position ranges from [30,000 to 100,000] based on Generate key, according to the case combination.
7. In every shifting operation, one bit is appended.
8. The second stage is to perform the multiplication operation between key and generate key.
9. The third stage is to perform the addition operation between the result of the second performed operation and chunk data.
10. The result of the third operation is becoming the chunk data (cipher text).
11. The above three operations perform each layer in three-layered encryption.
12. After assigning case's combination of each chunk, cases are incremented.
13. Finally, the chunks are stored.

The first and foremost steps are to convert the received or uploaded file to a binary form and sliced into blocks. The block size is fixed as 360,000 bits. The no of blocks depends upon the size of the binary plaintext. After the blocks are created, each block is intended in a three-layered encryption process. Each block has its case combination for both case1 and case 2, the case1 combination is used to reflect the keys priority list and the case 2 combination is used to reflect the generated keys priority list. Now a block has a priority list of both keys and generated keys to perform the three-layer of encryption. According to the priority list, each layer has its respective key and generate key.

The operation performed in each encryption layer is (i) right shift operation (ii) multiplication operation (iii) addition operation. The right shift operation range is between 30,000 and 100,000 and the range is fixed based on the generate key. After every right shift operation, append one bit in the first position of the block value as '1′ for a constant. The constant value helps to get the same value while decrypting. The multiplication operation is to be performed between key and generate key to avoid the looping process. Finally, addition operation is performed between the result of multiplication operation and numerically converted block value. The result of the addition operation in the numeric form immediately converts to the binary form. The binary form is used for further encryption layers.

After the operations are performed in three layers, it results in an encrypted block. The first encrypted block has named the file itself. The other encrypted block is an auto name. The first encrypted holds the second encrypted block name. The second encrypted block holds the other block name up to the final block. The block name is used for future work. Now the block is stored in the specified storage.

3.3. Decryption process

1. First, get the original file named chunk from the storage and it will undergo a three-layered decryption process.
2. Each layer of the decryption process uses the respective key and generate keys, according to the case's combination.
3. Each layer performs the left shift operation. The shift position range between [30,000 and 100,000] based on generate keys, according to the case combination.
4. In every shifting operation, one bit is reduced.
5. The first stage is to perform the multiplication operation between key and generate keys.
6. The second stage is to perform the subtraction operation between the result of the second performed operation and chunk data.
7. The third stage is to perform the left shift operation. The result of the third operation is a plaintext chunk and will get the next file name's to retrieve the original file.
8. Get other chunk files mentioned in the first file from the storage.
9. Above mentioned three-layered decryption is applicable for all chunks, until reaching the final chunk.
10. After assigning the case's combination of each chunk, cases are incremented to get the next chunk case's combination.
11. After completing the decryption process all chunks are merged and it results in the original file.
12. The file type of the original file is retrieved from the user key and stored in original file.

In the decryption process the original file named block is processed first that has undergone a three-layered decryption process. Case combination procedure has followed the same as encryption procedure. So the respective key and generate key is got to the decryption process for each layer and as well as each block. The operations in each decryption layer are (i) left shift operation (ii) multiplication operation (iii) subtraction operation. The left shift operation first deletes the constant value appended during the encryption and gets the range of shifting value, following the same procedure as in encryption. After getting the range, it performs the left shift. Then multiplication operation is performed between key and generate key. Finally, the subtraction operation is performed between the result of multiplication operation and numerically converted block value. The result of the subtraction operation in the numeric form immediately converts to the binary form.

After the completion of decryption process, it results plaintext block. The first plaintext block holds the second block name and the

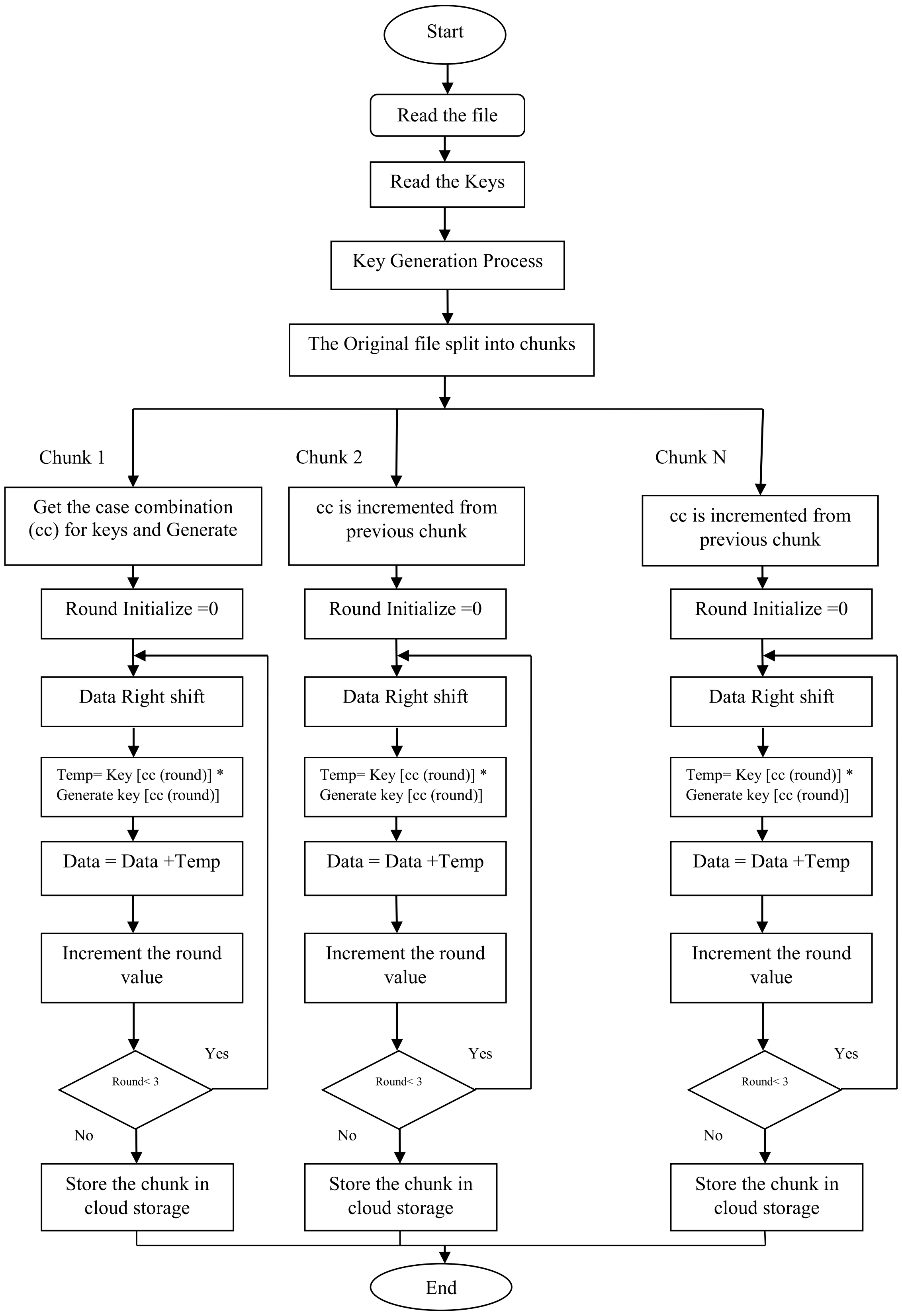
Start
Read the file
Read the Keys
Key Generation Process
The Original file split into chunks
Chunk 1
Chunk 2
Chunk N
Get the case combination (cc) for keys and Generate
cc is incremented from previous chunk
cc is incremented from previous chunk
Round Initialize =0
Data Right shift
Temp= Key [cc (round)] * Generate key [cc (round)]
Data = Data +Temp
Increment the round value
Round< 3
Yes
No
Store the chunk in cloud storage
End

Fig. 1. Flowchart for data encryption process.

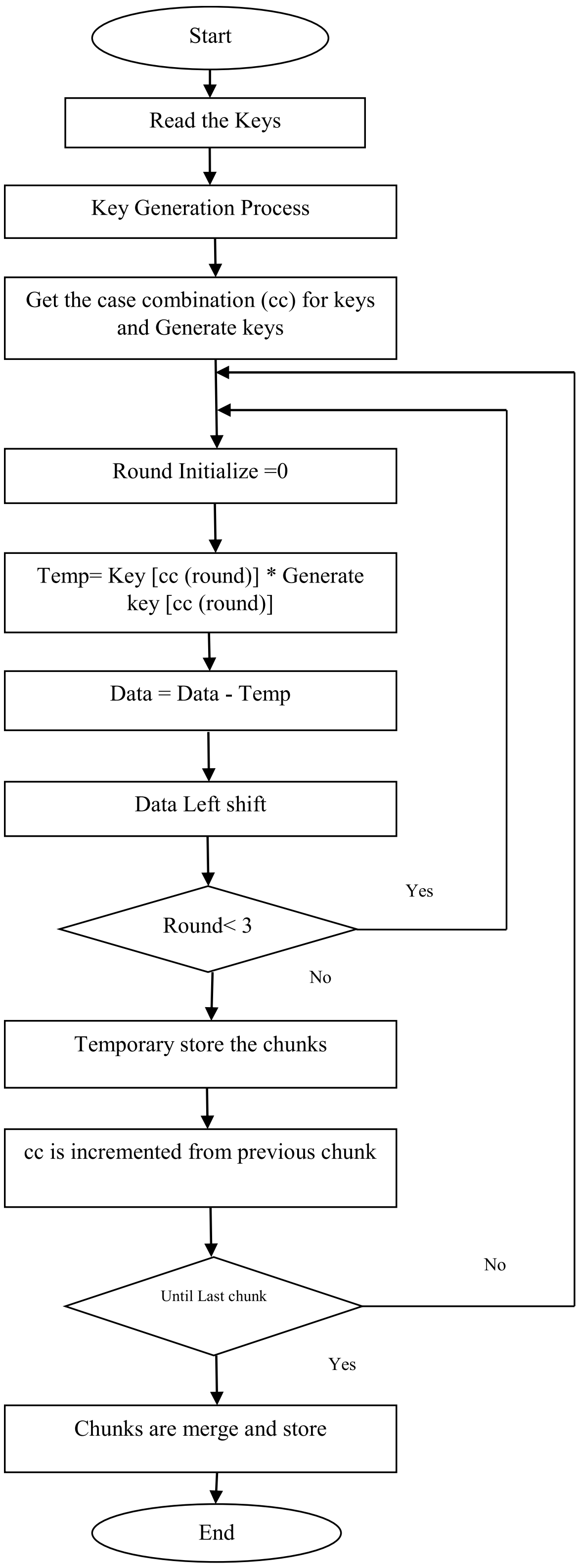
Start
Read the Keys
Key Generation Process
Get the case combination (cc) for keys and Generate keys
Round Initialize =0
Temp= Key [cc (round)] * Generate key [cc (round)]
Data = Data - Temp
Data Left shift
Round< 3
Yes
No
Temporary store the chunks
cc is incremented from previous chunk
Until Last chunk
No
Yes
Chunks are merge and store
End

Fig. 2. Flowchart for data decryption process.

Table 1
Performance results of UK algorithm.

Sl. No	FILE TYPE	SUB TYPE	SIZE (bytes)	ENCRYPTION TIME(seconds)	DECRYPTION TIME(seconds)
1	Document	xls	20,480	0.289	3.89
2	Document	doc	204,288	2.19	7.95
3	Document	doc	1,028,096	12.9	19.78
4	Document	pdf	5,222,400	72.69	84.98
5	Image	jpg	24,576	0.257	4.09
6	Image	png	212,992	2.25	7.53
7	Image	jpg	1,044,480	13.7	20.61
8	Image	jpg	6,053,888	73.546	86.529
9	Audio	mp3	20,480	.293	4.11
10	Audio	mp3	208,896	2.36	8.1
11	Audio	wav	1,073,152	13.43	20.89
12	Audio	mp3	5,289,384	74.056	85.956
13	Video	mp4	21,450	.246	4.01
14	Video	mp4	211,796	2.29	7.53
15	Video	mp4	1,150,976	13.6	21.01
16	Video	mp4	5,253,880	73.489	86.365

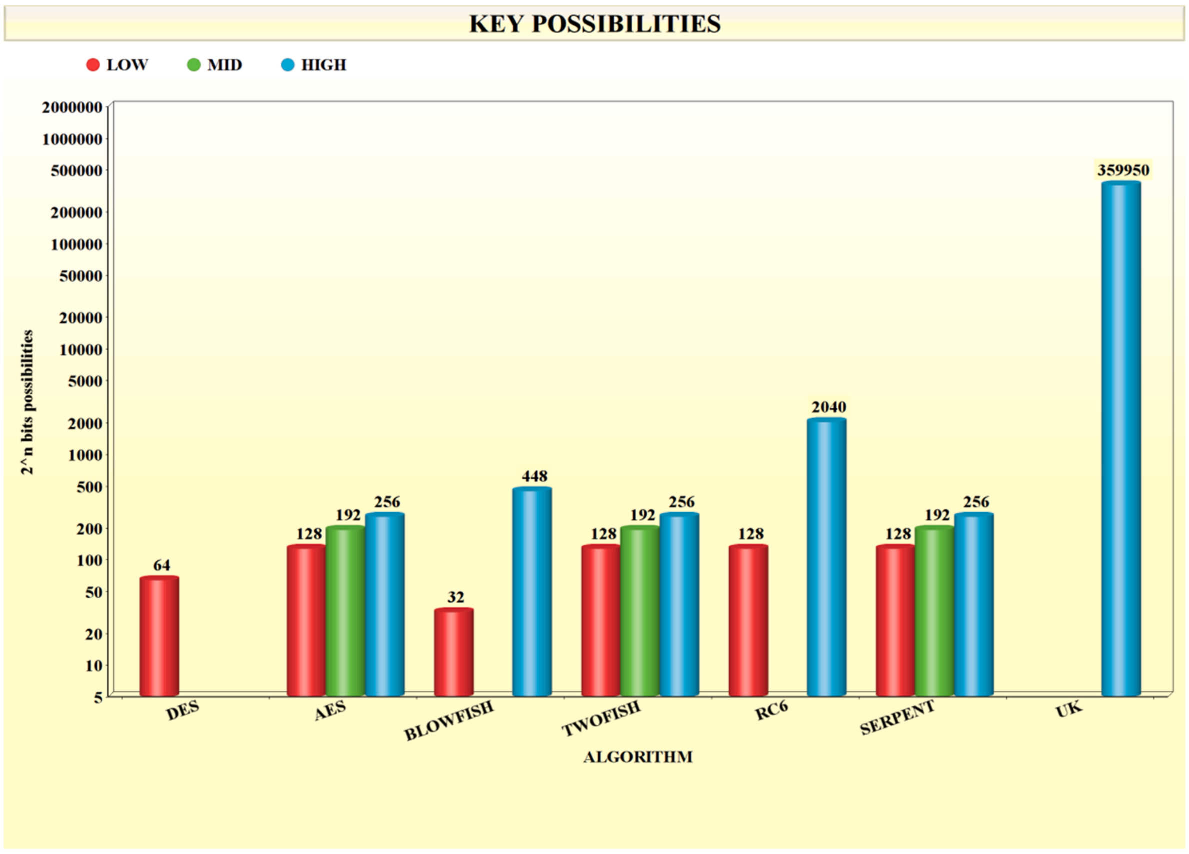

Fig. 3. 2ˆn key possibilities of encryption algorithm.

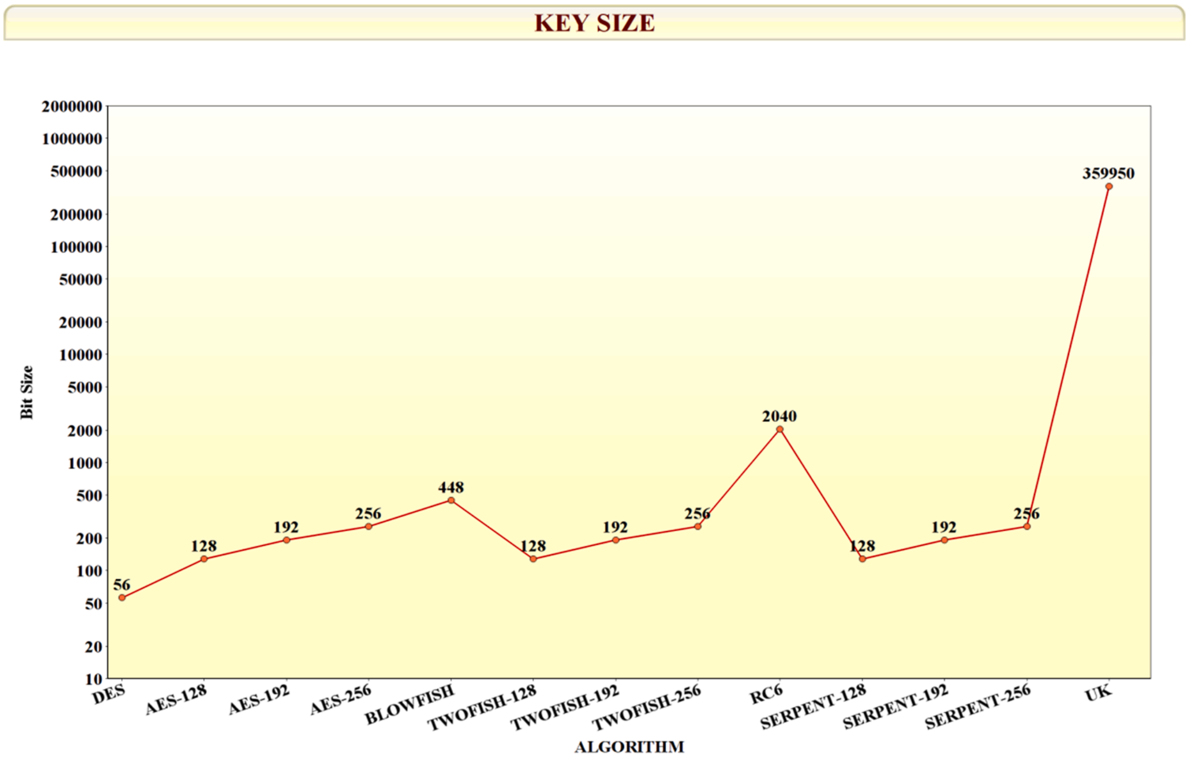

Fig. 4. Key size of encryption algorithm.

second block holds the rest of the blocks. After completing the decryption of all blocks, the block is merged. Finally, the merged block is converted to the file type as the fifth key.

$$CC1 = Sum(Length[Key_1] + Length[Key_2] + Length[Key_3]) \tag{1}$$

$$CC2 = Sum(\ Generated\_Key_s) \tag{2}$$

$$chunk[data] = \sum_{r=0}^{3} \ Swap[data] + \left(Key_{CC1[r]} * Generated\_Key_{CC2[r]}\right) \tag{3}$$

Eq. (1) represents the formula to get the first case combination (CC1) and Eq. (2) represents the formula to get the second Case Combination (CC2). Both CC1 and CC2 play a major role to create confusion in the plaintext block and they are used during the encryption and decryption. Eq. (3) represents the formation of the encryption procedure. Here the data undergoes three rounds (r), 0 to 3. Swap operation is enabled to perform the right shift for encryption and the left shift for decryption based on CC2. The multiplication operation is performed between the key as per CC1 and generated key as per CC2 based on round count. Finally, an addition operation is performed between swapped data and the result of the multiplication operation. The result of round1 data is an input of round2 and similarly it continues till the last round.

$$chunk[data] = \sum_{r=3}^{0} \ Swap[data] - \left(Key_{CC1[r]} * Generated\_Key_{CC2[r]}\right) \tag{4}$$

$$original\ data = join\left(chunk\ \left[\ data\ _{(0\ to\ n)}\ \right]\right) \tag{5}$$

Eq. (4) represents the formation of the decryption procedure. Here the data undergoes three rounds (r) in reverse, 3 to 0. The swap and multiplication are as same as discussed above. Finally performing the subtraction operation between swapped data and the result of the multiplication operation the original data gets after joining the decrypted chunks and it is represented in Eq. (5).

Fig. 1 flowchart states the data encryption process involved in the UK algorithm. It first reads the file to encrypt and reads the keys for encryption process. The key generation process is finished before the splitting process initiated. The chunks individually undergo the three layered encryption process and each layer has respective key depending upon the combinations. Each layer has right shift operation, multiplication operation and addition operation. After completing the three layered process the chunks are saved in storage.

Fig. 2 flowchart states the data decryption process involved in the UK algorithm. Reading the file, keys and key generation are as

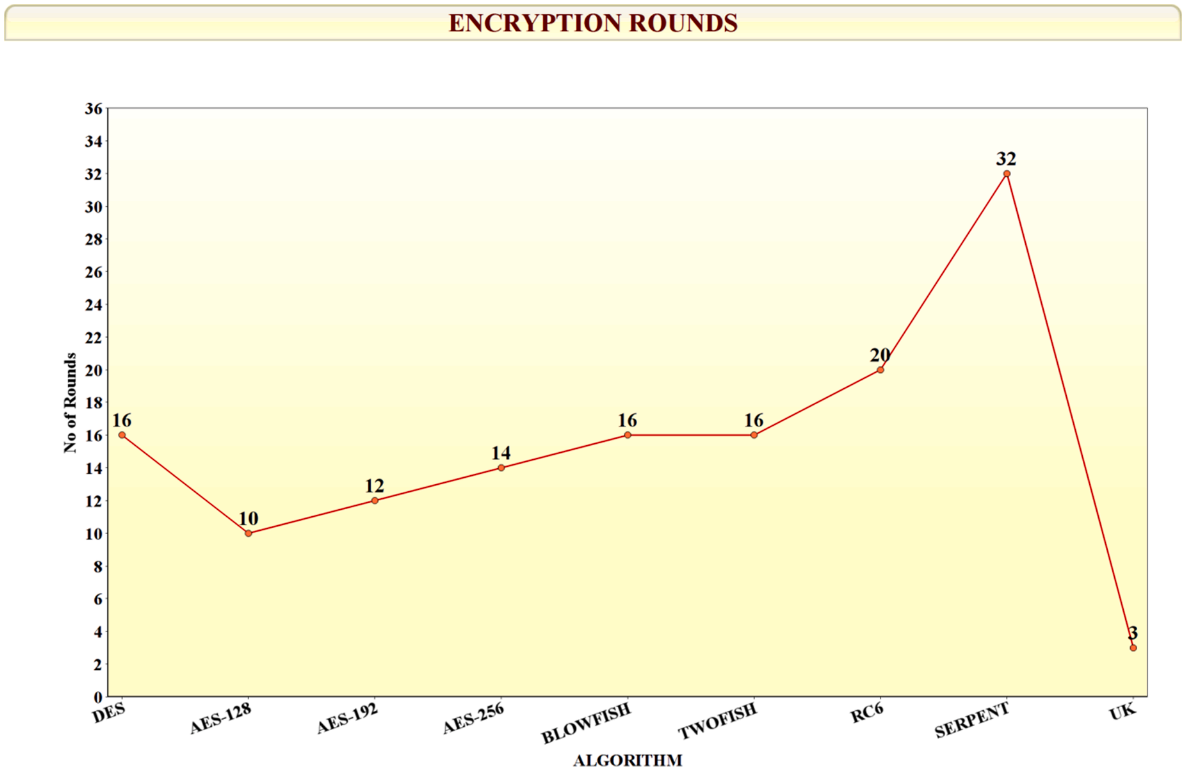

Fig. 5. Rounds of encryption algorithm.

same as encryption process. The chunks individually undergo the three layered decryption process and each layer has respective key depending upon the combinations. Each layer has multiplication operation, subtraction operation and Left shift operation. After completing the three layered process the Decrypted (original) data are saved in storage.

4. Results and discussion

To analyze the proposed UK algorithm efficiency by implementation on scenario-based concepts, the experimental execution is done in python language and the system configuration is Intel Core i7 (2.60 GHz) with 16 GB of RAM.

The proposed work architecture can be alternated to the conventional block cipher method. The present-64/128/192/256 block size encryptions are AES, DES, Triple DES, and RC5 but in the proposed UK algorithm method result block size 360,003 bits are more after the encryption process. The block size is invariable because of performing the arithmetic operation and adding the constant value during the encryption process. In the comparison between the conventional and UK method, the complexity of the block is very huge.

In the UK algorithm the block size is more than 360,000, it means even if the small phrase like “Hai how r u” is encrypted, the message is 11 characters and the bit size is 88 (11×8). The 88-bit plaintext is converted to 360,000 bits; and it is both positive and negative. In positive 88 bit message can hold the huge complexity 360,000, and brute force appears hard to break. In negative 88-bit message plaintext utilizing 360,000 bits need to be secured in terms of storage. If the file could be 5 MB, 10 MB it is very difficult for this kind of huge storage difference.

The key expansion technique is the operated unique because it is tailor-made for the UK algorithm. The interpretation itself has its own set of rules and regulations. For example the Character password must have a minimum of one binary character. The password strength is dependent upon using no binary character in regular intervals. These two rules help to result in invariable length keys while interpreting character passwords from the user. After the interpretation, the invariable key is expanded to the fixed key size 359,950. The key expansion has used the repetition method.

Table 1 shows that the UK algorithm can be handled in different file types like document, image, audio, and video. Here some lists of file type are tested: Excel, word, pdf, jpg, png, mp3, mp4. The Table 1 sl.no 1, 5,9,13 and 2, 6, 10, 14 are all almost similar in file size, encryption time and decryption time. But they are different in file type, so the table illustrates that encryption and decryption time is dependent on file size only not on file type. The table shows that the encryption time is comparatively lesser than the decryption time. In encryption process multi-threading is executed from first block but in decryption process multi-threading is executed from third block because of linked list concept. As explained in the proposed work section, first block holds the second block name and second block holds the rest of the block name, so after second block decryption only list of block name is gotten to decrypt and execute the multi-threading to finish the decryption quick.

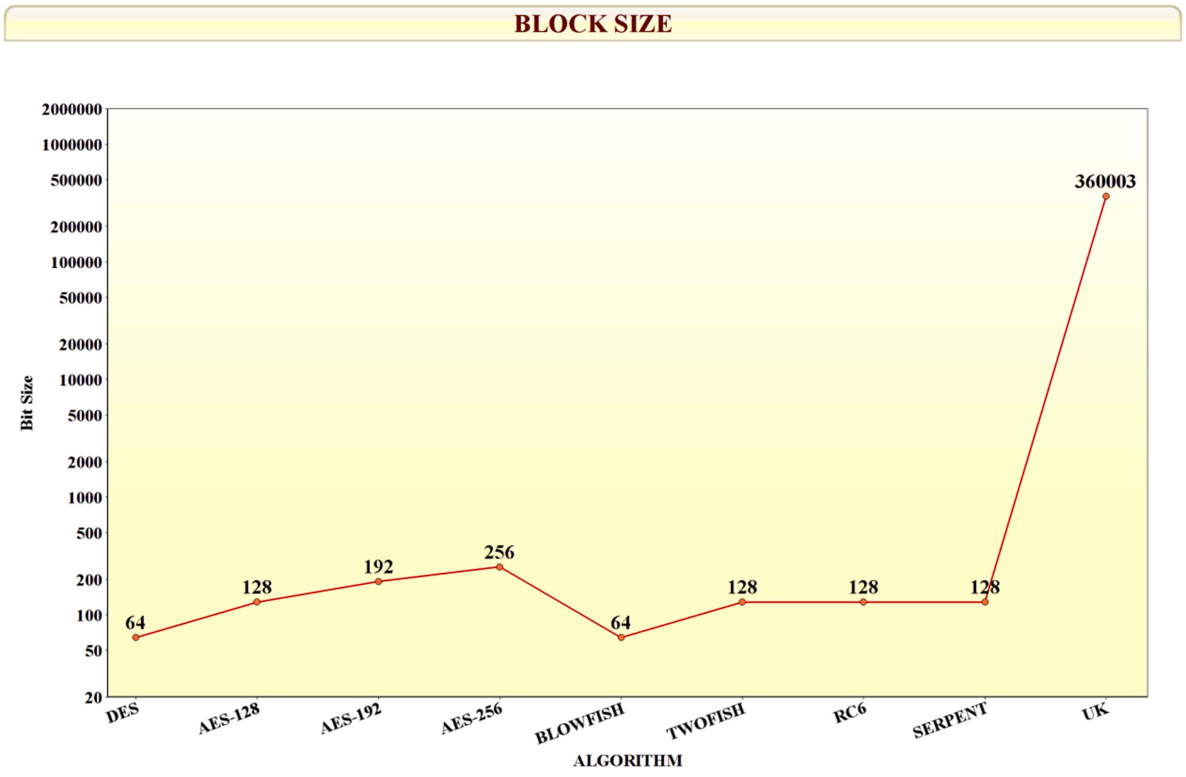

Fig. 6. Block sizes of encryption algorithm.

Table 2
Comparison of key size and key possibilities.

Algorithm	Key Size(bits)	Key possibilities	Rounds	Block size(bits)
DES	56	2^{56}	16	64
AES	128,192,256	2^{128},2^{196},2^{256}	10,12,14	128,192,256
BLOWFISH	32–448	2^{32} – 2^{448}	16	64
TWOFISH	128,192,256	2^{128},2^{196},2^{256}	16	128
RC6	128–2040	2^{128} – $2^{2,040}$	20	128
SERPENT	128,192,256	2^{128},2^{196},2^{256}	32	128
UK (proposed)	359,950	$2,^{359,950}$	3	360,003 + (depends on keys and data)

Fig. 3 represents the ’2′ power ’n’ number of possibilities in binary form key (i.e. 2ˆ32= 4,294,967,296) to brute force the encryption algorithms. The higher number of keys possibilities is proportionate to the high level of security of the keys and the brute force method is also difficult to crack key. Fig 3 states that the existing encryption algorithm has a maximum of 2040 possibilities in the RC6 algorithm and floatable from of 128 based on the user key input. Rest of the algorithms have lesser key possibilities. The proposed UK algorithm is far higher than the other encryption algorithms. It states a high level of security in keys possibilities.

Fig. 4 represents the keys size of an encryption algorithm and keys possibilities depending upon the size of the keys. So the largest key size creates a higher possibility for an encryption algorithm. An algorithm like AES, TWOFISH, and SERPENT work with different key sizes, and algorithms like BLOWFISH and RC6 are floatable key sizes based on user input. The fig shows that the larger key size is held by the UK algorithm.

Fig. 5 represents the number of rounds taken by the encryption algorithm. It shows the AES-128 holds a minimum of 10 rounds and the serpent holds a maximum of 32 rounds to perform the encryption operation. The proposed UK algorithm has 3 rounds only to perform encryption operations and it is feasible to increase the key size and block size for the encryption operation. Fig. 6 represents the block size of the encryption algorithms. The AES-256 holds the maximum block size of 256, and The DES, BLOWFISH holds the minimum block size of 64. The proposed UK algorithm is of huge block size compared with the existing algorithm as 360,003.

The difference between the existing and proposed methods is clearly seen in Table 2. The key size, key possibilities, and block size are very high when compared to the existing algorithms. The rounds are very less when compared to the existing algorithms. Here the block size is also not fixed, there is a minimum size of 360,003 but it may be increased to one or more bits depending upon the data and key actions in the encryption process. So predicting the block using the size is not possible. The proposed UK algorithm is entirely different from the existing algorithms, so the encryption & decryption time and complexity of block or data could not be compared.

5. Conclusion and future work

The paper is attempted to proves that the UK algorithm can handle numerous file formats, including documents, photos, audio files, and videos. To prove the same a relative number of files is inspected. It is observed that Excel, Word, PDF, JPG, PNG, MP3, and MP4 are turned to be acceptable file types. In terms of file size, encryption time, and decryption time, all file formats are fundamentally identical. However, based on the dissimilarity of the file kinds, the result shows that the time for encryption and decryption depends only on the file size and not on the file type. The difference in times in both encryption and decryption is significant and it is prominently shown in result discussion. The linked list notion causes the decryption process multi-threading to start from the third block which is quite contrary to the encryption process multi-threading starting from the first block. The first block stores the second block name, and the second block holds the remaining block names, as stated in the suggested work section. As a result, after the second block decryption a list of block names are received to decrypt and for which it becomes a must to use multi-threading to complete the decryption quickly.

Future work is to decrease the encryption and decryption time without compromising the security and high complexity. The space complexity is high in the proposed UK algorithm and needs to be optimistic. Implementing the UK algorithm in the user device environment to make the cost of maintaining the cloud storage low and evolve the mini, micro level Cloud Storage shall be attempted. In this case, the Cloud Storage provider does not need huge infrastructures, does not require computational devices to process the data and there is no need for separate data transmission to secure the data. Employing the encryption algorithm in the user environment helps to give full authority to the user on their data. The Cloud-Service provider holds only encrypted data of the user and with low-cost consumption, and this method is feasible.

Funding statement

The authors received no specific funding for this study.

Declaration of Competing Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Data availability

The data that has been used is confidential.

References

[1] Khan SS, Tuteja PRR. Security in cloud computing using cryptographic algorithms. Int J Innov Res Comput Commun Eng 2015;3(1):148–54. https://doi.org/10.15680/ijircce.2015.0301035 [Online]. Available: Copyright to IJIRCCE.
[2] Pandith MY. Data security and privacy concerns in cloud computing. Internet Things Cloud Comput 2014;2(2):6–11. https://doi.org/10.11648/j.iotcc.20140202.11.
[3] Smid ME. Development of the advanced encryption standard. J Res Natl Inst Stand Technol 2021;126(126024):1–18. https://doi.org/10.6028/jres.126.024.
[4] Rothke B. A look at the advanced encryption standard (AES), *Information security management handbook*. Auerbach Publications; 2007. p. 1151–8. https://doi.org/10.1201/9781439833032.ch89.
[5] Kumar TM, Karthigaikumar P. Implementation of a high-speed and high-throughput advanced encryption standard. Intell Autom Soft Comput 2022;31(2): 1025–36. https://doi.org/10.32604/iasc.2022.020090.
[6] Bharathi R, Parvatham N. Light-weight present block cipher model for IoT security on FPGA. Intell Autom Soft Comput 2022;33(1):35–49. https://doi.org/10.32604/iasc.2022.020681.
[7] Tahir M, Sardaraz M, Mehmood Z, Muhammad S. CryptoGA: a cryptosystem based on genetic algorithm for cloud data security. Clust Comput 2021;24(2): 739–52. https://doi.org/10.1007/s10586-020-03157-4.
[8] Ferdush J, Begum M, Uddin MS. Chaotic lightweight cryptosystem for image encryption. Adv Multimed 2021;2021(5527295). https://doi.org/10.1155/2021/5527295.
[9] Park T, Seo H, Lee S, Kim H. Secure data encryption for cloud-based human care services. J Sens 2018;2018(6492592). https://doi.org/10.1155/2018/6492592.
[10] Coppersmith D. The data encryption standard (DES) and its strength against attacks. IBM J Res Dev 1994;38(3):243–50. https://doi.org/10.1147/rd.383.0243. May.
[11] Schneier B. Description of a new variable-length key, 64-bit block cipher (Blow- fish. Fast Softw Encryption 1994;809:191–204. https://doi.org/10.1007/3-540-58108-1_24.
[12] Bajpai P, Enbody R. Dissecting. net ransomware: key generation, encryption and operation. Netw Secur 2020;2020(2):8–14. https://doi.org/10.1016/S1353-4858(20)30020-9.
[13] Panda M. Performance analysis of encryption algorithms for security. In: Proceedings of the international conference on signal processing, communication, power and embedded system (SCOPES); 2016. p. 278–84. https://doi.org/10.1109/SCOPES.2016.7955835.
[14] Song Y, Wang H, Wei X, Wu L. Efficient attribute-based encryption with privacy-preserving key generation and its application in industrial cloud. Secur Commun Netw 2019;2019(3249726). https://doi.org/10.1155/2019/3249726.
[15] Mudepalli S, Rao VS, Kumar RK. An efficient data retrieval approach using blowfish encryption on cloud ciphertext retrieval in cloud computing. In: Proceedings of the international conference on intelligent computing and control systems (ICICCS); 2017. p. 267–71. https://doi.org/10.1109/ICCONS.2017.8250724.

[16] Haq TU, Shah T, Siddiqui GF, Iqbal MZ, Hameed IA, Jamil H. Improved twofish algorithm: a digital image enciphering application. IEEE Access 2021;9:76518. https://doi.org/10.1109/ACCESS.2021.3081792. –30.
[17] Barman S, Chattopadhyay S, Samanta D, Panchal G. A novel secure key-exchange protocol using biometrics of the sender and receiver. Comput Electr Eng 2017; 64:65–82. https://doi.org/10.1016/J.COMPELECENG.2016.11.017.
[18] Krishnaveni S, Prabhakaran S, Sivamohan S, Sridhar S. Network intrusion detection based on ensemble classification and feature selection method for cloud computing. Concurr Comput Pract Exp 2022:1–29. https://doi.org/10.1002/cpe.6838.
[19] Das D, Sethuraman SC, Satapathy SC. A decentralized open web cryptographic standard. Comput Electr Eng 2022;99:107751. https://doi.org/10.1016/J.COMPELECENG.2022.107751.
[20] Atiewi S, Al-Rahayfeh A, Almiani M, Yussof S, Alfandi O, et al. Scalable and secure big data IoT system based on multifactor authentication and lightweight cryptography. IEEE Access 2020;8:113498–511. https://doi.org/10.1109/ACCESS.2020.3002815.

B. UMAPATHY received an M.Phil; degree in computer science from VELS University, Chennai, India, in 2015. He is pursuing his Ph.D; degree in Computer Science at SRM IST, Chennai, India, since 2020. His-research interest includes Encryption and Decryption, Cloud Security, and Cryptography.

G. Kalpana has completed her Ph.D; from SRM IST, Chennai. Her Current research focuses on parallel and Distributed Computing, Block Chain Technology, and Cloud Security. Currently, she is working as an Associate Professor and Head in the Department of Computer Science, CSH, SRM IST. She has published more than 15 papers in journals and conferences. She has also filed an Indian patent in the year 2021.