Enhanced Symmetric Encryption Technique for Securing Users' Data in Public Cloud Environment

A. Fairosebanu[1†], Dr. A. Nisha Jebaseeli[2††],

Government Arts and Science College, Kumulur, Lalgudi, Affiliated to Bharathidasan University, Trichy, India

Summary

Cloud is the biggest backbone for storing data and maintaining the data reliably. Cloud helps many enterprises to keep their data alive without any management issues. Most enterprises are like to have their data in the cloud because of its benefits. It has more compatibility for accessing the data. Even though the cloud gives many advantages to the users, it has the biggest problem in maintaining the data securely. Cloud is not transparent. Users can't know the data storage details, and the administrators maintain and monitor the data. The public nature of the cloud may disclose data to other users or the possibility that the administrators can view the data. Hence, data security is the biggest challenge in the cloud environment. To handle the security challenges in the cloud, cryptography approaches may help to do. This paper enhances cryptography data security techniques based on encryption. The research focuses on the symmetric nature of encryption because of its capability to process a huge amount of data. The proposed approach is block cipher encryption to encrypt and decrypt the data using a key. The proposed technique is tested for its security efficiency and performance using the cloud-based deployment and security analysis tool. Test results demonstrate that the proposed approach is more efficient in the public cloud environment.

Keywords:

Cloud Computing, Cryptography, Encryption, Symmetric Approach, Block Cipher.

1. Introduction

Cloud computing is an evolutionary technology that derives many concepts from the existing technologies like parallel computing, pervasive computing, utility computing and grid computing. It is internet-based computing. Without an internet connection, it may not be possible to access the cloud [1]. It can be delivered as a service to the user based on their computing requirement. Users can be general users or enterprises. According to the user's requirement, they can access the cloud [2]. The cloud provisioned the users to get more benefits from their virtual nature. It provides a collaborative environment for working together in a single environment. The cloud helps the small and medium scale industries improve their IT business to handle it properly [3]. The cloud's main objective is to provide complete computing services, especially on providing storage. Cloud provides a huge amount of virtual storage to maintain a large amount of data. Due to the unlimited nature of the cloud, users can get limitless services for all computing services. Enterprises are adopting cloud for their storage outsourcing [4].

Apart from all cloud benefits and advantages, the cloud has the biggest fall in securing data in storage. Figure 1 shows the challenges in the cloud [5]. Security is the biggest concern in the cloud. The cryptography techniques generally address security. Cryptography can be delivered in the different security services. The list of cryptography security services is Authentication, Authorization, Confidentiality, Integrity and Availability [6]. The security service ensures the level of security at different levels. The authentication ensures the originality of the incoming users. Authorization talks about the authorized user roles in the network or enterprises. Confidentiality is the topmost concern in security to ensure that the authorized users only access the data. Integrity tells whether the data is modified or not during the transmission. Availability talks about the existence of Deny of service in the network [7]. Among all the services in cryptography, confidentiality needs more attention to maintain the data securely. Confidentiality of the data is maintained by using encryption techniques.

Figure – 1 Security Risk and Challenges

The encryption techniques are divided into two categories symmetric and asymmetric. The asymmetric

Manuscript received April 5, 2022
Manuscript revised April 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.4.93

may provide more security than the symmetric, but it can't be used in a large amount of data encryption. It can be only used for password encryption and other secret code encryption. Symmetric encryption is a more useful technique in addressing the confidentiality of the data stored in the cloud. Symmetric encryption uses a unique key to encrypt and decrypt. The sender encrypts the data using a key, and the key is securely communicated to the receiver to retrieve the data. The proposed encryption technique in the paper is based on the cryptosystem's symmetric nature, and it is block cipher encryption [8]. It can encrypt a block of data at a time. The proposed encryption technique is enhanced from the present techniques concerning the number of rounds and keys used in the encryption. The key is a block of bits that can be processed to generate the necessary sub-version according to the round. The enhanced cipher is used in the public cloud environment to secure users' data.

2. Related Works

Cloud is used for storing huge amounts of data storage. When connected to the cloud, we are also ready to face the biggest challenges from data security. Cryptography techniques can address security issues. Ansar et al. [9] suggested that a single security algorithm to secure data in the cloud is not enough. Ansar et al. delivered a mechanism to use symmetric key cryptography and steganography techniques. In their methodology, the file is separated into eight chunks. Each chunk is encrypted using a different cryptography algorithm. Key is used as a stegno image, and it is shared with the recipient to retrieve the data. The authors suggested using more than five encryption algorithms to encrypt different chunks. However, they failed to describe whether they use the same key for all algorithms or different keys for each file chunk. To enhance the data security in the cloud, Gangireddy et al. [10] enhanced the blowfish algorithm and used the k-medoid clustering to cluster the secret information. The dragonfly approach is implemented in their work to improve clustering accuracy. However, the author does not enlighten how they enhance the blowfish algorithm from the research perspective. Similarly, Thangapandiyan et al. [11] proposed a modified elliptic curve downward cryptography. A verification id is assigned to each user and admin. Modification of the ECC is not given in the paper. Pushpa [12] proposed a hybrid approach encryption technique to improve medical data security in the cloud environment. The paper combines the blowfish and two fish algorithms to enhance security in the cloud. Sanjeev et al. [13] proposed a multilevel hybrid data security approach to strengthening data security in the cloud. The authors proposed to use symmetric and asymmetric cryptography algorithms in a multilevel manner. According to the methodology, the data is first encrypted using the DES algorithm, and the encrypted data from the DES is again encrypted using the RSA algorithm. This hybrid approach takes more time for encryption and decryption. Hosseinet al. [14] suggested avoiding using plex cryptography encryption algorithms for secure data in the cloud before the speed of data encryption is more significant in the cloud. Hence, the author developed a solution that uses an improved blowfish algorithm to encrypt the data and the elliptic curve algorithm to encrypt the key used in the encryption. Along with this, using a digital signature to ensure the integrity of the data. Many more literature [15]-[20] are reviewed. Most of the literature is described their proposed approach in the manner of hybrid way. But, combining two or more algorithms into a single algorithm must consider the time taken for processing data before uploading to the cloud.

3. Methodology

The proposed encryption technique is an enhanced standard of existing cryptography approaches. A block cipher encryption uses a secret key to encrypt and decrypt the data. In the existing encryption approaches, the number of rounds executed for encryption and decryption is fixed and known to adversarial users. The hacker can try to hack the data by knowing the number of rounds in the encryption. The proposed technique is enhanced by hiding the number of rounds executed for encryption and decryption. The number of rounds of the encryption is dynamically decided from the key. The actual key used for this proposed encryption technique is 196-bit. But, at the time of generating the key, it is 200-bit. The last 8 bits are split into two 4-bit and find XoR between those two 4-bits to get a 4-bit. Hence the 200-bit is converted to a 196-bit key and used for encryption. The last four is taken as the sub-key1, which denotes the number of rounds the encryption is executed for a certain plaintext. In the same manner, the decryption is also executed. The encryption of the proposed technique takes 64-bit as input and produces 64-bit as output. The input plaintext data is processed in two cryptography techniques: substitution and transposition. Figure 2 depicts the block diagram of the proposed encryption technique.

4. Proposed PUCS Cipher

The proposed encryption PUCS Cipher encrypts the data as a 64-bit block. It is executed for the number of rounds based on the key. The rounds of the encryption are not fixed; it is changed for different input data. The key used for encryption is 196-bit. It is processed and split into

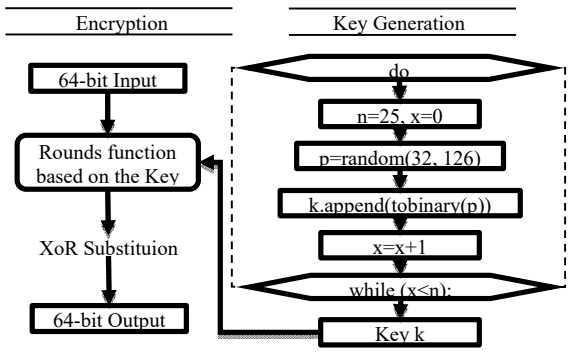

Figure – 2 Block Diagram of Encryption and Key Generation in PUCS Cipher

four subkeys. The plaintext bits are permutated based on subkey 2.Permutation denotes the transposition of bits. After permutation, the 64-bit is split into two equal half of 32-bit. For example, subkey 3 is 64-bit and split into two 32-bit. The two 32-bit is XoRed with subkey 3. After XoR, the two 32-bit is swapped. After swapping, two 32-bit are merged into 64 bits, and now a single round is completed. The same round of execution is continued at the number of times denoted by subkey 1. Once all rounds are completed, the 64-bit is XoRed with subkey 4 and generates the 64-bit plaintext. Figure 3 shows the encryption execution of the proposed technique.

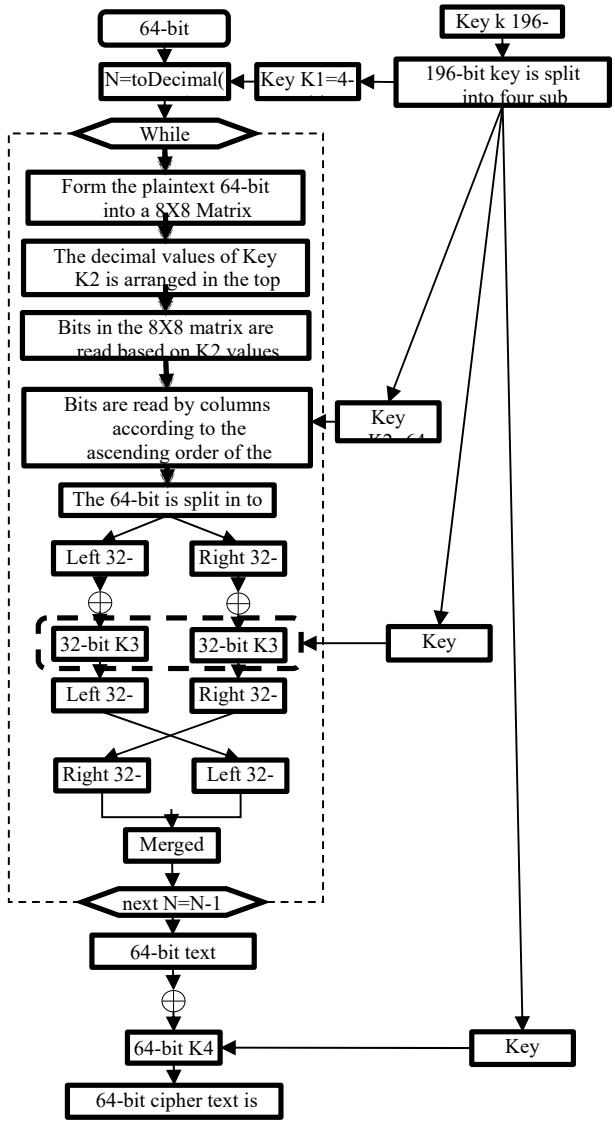

Figure – 3 PUCS Cipher Encryption Procedure

4.1 Procedure of PUCS Cipher

Step 1: Users' data are taken as Input Plain Text (PTEXT)
Step 2: Consider the binaries of PTEXT
Step 3: Input PT is divided into 64 bits blocks. PUCS Cipher encrypts 64-bit blocks at a time.
Step 4: Get a 196-bit key KEY for PUCS Cipher from KPMaaS.
Step 5: Last four bits in the key KEY denotes number rounds to be executed for encrypting the data.
Step 6: The round function starts. Form the PTEXT into an 8X8 Matrix MAT.
Step 7: Get the first 64-bit subkey SKEY1 from the 196-bit key.
Step 8: Convert the SKEY1 64-bit into corresponding eight decimal values.
Step 9: Arrange the eight decimals on the top of each eight-column of the MAT.
Step 10: Read the bits from the MAT by column based on the ascending order of the eight decimal values placed on the top of each column.
Step 11: The 64-bit is split into two equal half of 32-bit blocks by reading even and odd positional bits separately.
Step 12: Get the second 64-bit subkey SKEY2 from 196-bit KEY.
Step 13: Split the SKEY2 into two 32-bit keys.
Step 14: Find the XoR of two 32-bit plaintexts with two 32-bit keys and get the result of two 32-bits blocks.
Step 15: 32-bit swap is carried out.
Step 16: Merge the resulting two 32-bit blocks into 64-bit by alternatively placing bits from both blocks.
Step 17: The round function is completed. Steps from step 6 to step 15 is repeated in several rounds based on the encryption rounds. The result from the first round is given as the input to the next round.
Step 18: After all rounds, a 64-bit output is derived. It is XoR with the third subkey SKEY3 from the key K.
Step 19: The resulting 64 bits from Step 17 is the cipher text CTEXT.

5. Experiment with Sample Data

The proposed technique has experimented with sample plaintext shown in the box below.

Step 1: Users' data are taken as Input Plain Text (PTEXT)

Table – 1 Sample Input Data of User

PName	***PMR No***	DOB	Hospital Name	Disease	Amount
Ram K	SU9322	19/8/1987	suthan	hunger	19000

P_{TEXT}→ RamKSU932219/8/1987suthanhunger19000

P_{TEXT} converted to Decimal:

R→82 a→97 m→10 K→75 S→83 U→85
9→57 3→51 2→50 2→50 1→49
9→57 /→47 8→56 /→47 1→49
9→57 8→567→55 s→115 u→117
t→116 h→104 a→97 n→110h→104
u→117 n→110 g→103 e→101
1→49 9→57 0→48 0→48 0→48

Step 2 : Consider the binaries of P_{TEXT}

P_{TEXT} converted to Binaries:

01010010 01100001 01101101 01001011 01010011
01010101 00111001 00110011 00110010 00110010
00110001 00111001 00101111 00111000 00111001
00110010 00111001 00111000 00110111 01110011
01110101 01110100 01101000 01100001
01101110 01101000 01110101 01101110 01100111
01100101 01110010 00110001 00110010 00110000
00110000 00110000

Step 3: Input PT is divided into 64 bits blocks. PUCS Cipher encrypts 64 bits blocks. Experiment considers first 64 bits block for encryption:

P_{TEXT}→01010010 01100001 01101101 01001011 01010011 01010101 00111001 00110011

Step 4: Get a 196 bits key for PUCS Cipher from KPMaaS.

K_{EY}→M{WO]GXB 4k0hGGP`]'+mG^2L 3

Decimal equivalent of K_{EY}→ 77 123 87 79 93 71 88 66 52 107 48 104 71 71 80 96 93 39 43 109 71 94 50 76 58

Binaries of K_{EY}→01001101 01111011 01010111 01001111
01011101 01000111 01011000 01000010 00110100 01101011
00110000 01101000 01000111 01000111
01010000 01100000 01011101 00100111 00101011 01101101
01000111 01011110 00110010 01001100 0011

KEY Size: 196 Bits.

Step 5: Last four bits in the key K_{EY} denotes number rounds to be executed for encrypting the data.

Last four bits→0011→3

Number of Rounds→3

Step 6: The round function starts. Form the P_{TEXT} into an 8_X8 Matrix M_{AT}.

Round 1

0	1	0	1	0	0	1	0
0	1	1	0	0	0	0	1
0	1	1	0	1	1	0	1
0	1	0	0	1	0	1	1
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1
0	0	1	1	1	0	0	1
0	0	1	1	0	0	1	1

Step 7: Get the first 64-bit as a subkey SK_{EY1} from 196 bits key.

SK_{EY1} →01001101 01111011 01010111 01001111 01011101 01000111 01011000 01000010

Step 8: Convert the SK_{EY1} 64-bit into corresponding eight decimal values.

r→114 SK_{EY1} →77 123 87 79 93 71 88 66

Step 9: Arrange the eight decimals on the top of each eight-column of the M_{AT}.

77	**123**	**87**	**79**	**93**	**71**	**88**	**66**
0	1	0	1	0	0	1	0
0	1	1	0	0	0	0	1
0	1	1	0	1	1	0	1
0	1	0	0	1	0	1	1
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1
0	0	1	1	1	0	0	1
0	0	1	1	0	0	1	1

Step 10: Read the bits from the M_{AT} by column based on the ascending order of the eight decimal values placed on the top of each column.

P_{TEXT}→01111111 00100100 00000000 10001111 01100011 10011001 00110010 11111100

Step 11: The 64-bit is split into two equal half of 32-bit blocks by reading even and odd positional bits separately.

OBlock→01110100 00001011 01011010 01011110

EBlock→11110010 00111001 10010101 01001110

Step 12: Get the second 64-bit Subkey SK_{EY2} from 196-bit $K_{EY.}$

SK_{EY2} →00110100 01101011 00110000 01101000 01000111 01000111 01010000 01100000

Step 13: Split the SK_{EY2} into two 32-bit keys.

LSK_{EY2}→00110100 01101011 00110000 01101000

RSK_{EY2}→01000111 01000111 01010000 01100000

Step 14: Find the XoR of two 32-bit plaintexts with two 32-bit keys and get the result of two 32-bit blocks.

OBlock→01110100 00001011 01011010 01011110
LSK_{EY2}→00110100 01101011 00110000 01101000
XoR01~~00~~0000 01100000 01101010 00110110
Output

EBlock→11110010 00111001 10010101 01001110
RSK_{EY2}→01000111 01000111 01010000 01100000
XoR Output→10110101 01111110 11000101 00101110

Step 15: Merge the resulting two 32-bit blocks into 64-bit by alternatively placing bits from both blocks.

P_{TEXT}→01100101 00010001 00111101 01010100 01111000 10011001 00001110 01111100

Step 16: The round function is completed. Steps from Step 6 to Step 15 is repeated in several rounds based on the encryption rounds. The result from the first round is given as the input to the next round.

The experiment considers the plaintext with 64-bit, and it is executed for one round based on the key generated for this plaintext.

6. Implementation Setup

The proposed work is implemented and tested in the cloud environment. The proposed encryption procedure is coded in c#.net and developed as a web-based application. The developed application is hosted in the cloud environment called MyASP.NET. The MyASP.NET is a cloud-based platform providing service. They provide a platform to host the user's application. The environment allows only the application that is developed in the .NET framework. The proposed technique is developed in the Visual studio 2012 .Net framework. The hosting environment provides a user-friendly environment to host thedeveloped application. The hosting environment is the cloud server. Once the applicationis hosted in the MyASP.NET platform service, it can be accessed by any user from anywhere in the world. MyASP.NET environment provides a domain name for the application hosted in the cloud server. Users can use the domain name of the hosting application to open it. The client can use any operating system to access the hosted application. The developed application is tested for its performance according to the time taken for encryption and decryption. First, the application is provisioned with upload the plaintext to the server. Now the plaintext is encrypted using the proposed encryption technique. Finally, the encrypted text is displayed in the corresponding place in the application. During this encryption, the time is calculated. The time is calculated from the MyASP.NET server. Similarly, other encryption techniques test their performance based on the time taken for encryption and decryption. The performance and security level result is compared in the following section.

7. Results and Discussions

The proposed technique and existing techniques are tested with a similar implementation setup. The performance of the proposed encryption is calculated by the time taken for encrypting and decrypting the data. The technique is tested with sample medical data. As discussed earlier, the proposed technique encrypts 64-bit plaintext at a time. According to this, the input given to the technique is 64bit from the medical data. Then, the technique encrypts and decrypts the data. The time taken is noted for different sizes of the data. The taken to encrypt the data varies based on the size of the key and the number of rounds executed for encrypting the data. The results shown in the following tables are taken from the average time taken for encrypting the data up to five to six encryption or decryption. The total number of rounds executes for encryption and decryption is 16 rounds. Table 2 shows the performance comparison based on encryption time.

Table – 2 Performance Comparison by Encryption Time

Size	***DES***	**Blowfish**	**PUCS Cipher**
100 KB	72	44	37
200 KB	141	85	75
300 KB	213	132	112
400 KB	282	177	150
500 KB	355	223	188

The technique's decryption time is also considered for measuring performance. Decryption takes a more or less similar time to encryption. Table 3 shows the decryption time comparison with the present techniques

Table – 3 Performance Comparison by Decryption Time

Size	***DES***	**Blowfish**	**PUCS Cipher**
100 KB	69	42	31
200 KB	139	81	64
300 KB	207	128	103
400 KB	276	173	138
500 KB	350	219	169

The time taken varies according to the changes in the execution of rounds. Moreover, the proposed techniques are not run simultaneously in the same rounds. Instead, the round of execution is decided based on the key.

Therefore, encrypting 100KB of data with one round is approximately 2.33 milliseconds. Table 4 shows the time comparison of the encryption of the proposed technique with different rounds

Table – 4 Comparison of the Encryption of the Proposed Technique with Different Rounds

Rounds / ***Size***	*5 Rounds*	8 Rounds	10 Rounds	15 Rounds
100 KB	12	19	23	35
200 KB	25	37	46	69
300 KB	35	58	70	106
400 KB	48	75	92	141
500 KB	61	96	116	173

The more important measure of the encryption technique's efficiency is security strength. Based on the percentage of security level only, it can be claimed that the proposed technique is more efficient than the existing techniques. In this contact, the proposed technique is tested for security level using a hacking tool called ABC universal hacking tool. This tool is used to analyze the security strength of the encryption techniques. Furthermore, the tool hacks the encrypted data generated by the proposed techniques and tries to get the plaintext without knowing the key. The percentage of security is calculated from the percentage of plaintext found by the tool. Finally, the retrieved plaintext by the tool is matched with the original plaintext based on the percentage of matching the security level. Table 5 show the security level of all encryption techniques considered in the research.

Table – 5 Percentage of Security Level

S.No.	***Encryption Techniques***	**Security Level (%)**
1	DES	78
2	Blowfish	84
3	PUCS Cipher	89

The results shown in the tables show that the proposed encryption is more efficient to perform in the public cloud. Furthermore, performance comparison shows that compared to other approaches, the proposed technique perform well and execute the data in minimum time duration than the existing approaches. The proposed approach is measured for its efficiency in securing the data stored in the cloud. The table result shows that the security level of the proposed technique is higher than the existing security techniques.

8. Conclusion

Cloud implementation is more difficult with security approaches. The proposed technique is the block cipher encryption technique enhanced from the present technique. The enhancement is done in rounds, round function and key used. The proposed technique is 196-bit encryption. It is executed according to the keys for the number of rounds. The number of round execution is not fixed in the proposed technique. The proposed technique is tested for its efficiency with existing techniques for its performance and security level. Compared to the existing technique, the proposed technique performs well in executing the data in minimum duration. The proposed technique's security level is also higher than the existing approaches. The security level is tested using the security analyst tool called Hackman tool.

References

[1] K. B. Sarmila and S. V. Manisekaran, A Study on Security Considerations in IoT Environment and Data Protection Methodologies for Communication in Cloud Computing, IEEE International Carnahan Conference on Security Technology (ICCST), 2019, pp. 1-6.

[2] R. Doshi and V. Kute, A Review Paper on Security Concerns in Cloud Computing and Proposed Security Models, IEEE International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-4.

[3] C. Myeonggil, The Security Risks of Cloud Computing,IEEE International Conference on Computational Science and Engineering, 2019, pp. 330-330.

[4] P. Hima Bindu1, T. Bhaskar Reddy2, *An Exploration Of Security Issues For Cloud Computing*, Journal of Engineering and Science, Vol 11, Issue 1, 2020, pp. 144-152.

[5] New 2019 Cloud Security Research Reveals Key Challenges for Security Professionals, https://www.businesswire.com/news/home/20190624005387/en/New-2019-Cloud-Security-Research-Reveals-Key-Challenges-for-Security-Professionals, viewed March 2020.

[6] A. S. Mattoo, D. Upadhyay, A. K. Dubey and M. K. Shukla, An approach to analyze and protect data on Untrusted Cloud Network, IEEE International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2020, pp. 139-144.

[7] P. A. Pandire and V. B. Gaikwad, Attack Detection in Cloud Virtual Environment and Prevention Using Honeypot,International Conference on Inventive Research in Computing Applications (ICIRCA), 2018, pp. 515-520.

[8] H. Song, J. Li and H. Li, *A Cloud Secure Storage Mechanism Based on Data Dispersion and Encryption*, in IEEE Access, vol. 9, 2021, pp. 63745-63751.

[9] Ansar I. Sheikh, Twinkle Athole, KunalWankhede, ChandanKawle, Mona Kshirsagar, Data Security using Hybrid Cryptography in Cloud, Journal of Emerging Technologies and Innovative Research, Volume 7, Issue 5, 2020, pp. 56-61.

[10] VenkataKoti Reddy Gangireddy, SrihariKannan, KarthikSubburathinam, Implementation of enhanced blowfish algorithm in the cloudenvironment, Journal of Ambient Intelligence and Humanized Computing, Springer-Verlag, 2020, pp. 1-7.
[11] M. Thangapandiyan, P. M. RubeshAnand and K. Sakthidasan, Enhanced Cloud Security Implementation usingModified ECC Algorithm, IEEE International Conference on Communication and Signal Processing, April 3-5, 2018, pp. 1019-1022.
[12] B. Pushpa, Hybrid Data Encryption Algorithm for SecureMedical Data Transmission in Cloud Environment, IEEE International Conference on Computing Methodologies and Communication, 2020, pp. 329-334.
[13] Sanjeev Kumar, GarimaKarnani, Madhu Sharma Gaur, Anju Mishra, Cloud Security using Hybrid CryptographyAlgorithms, IEEE International Conference on Intelligent Engineering and Management, 2021, pp. 597-603.
[14] HosseinAbroshan, A Hybrid Encryption Solution to Improve Cloud Computing Security using Symmetric and Asymmetric Cryptography Algorithms, International Journal of Advanced Computer Science and Applications,Vol. 12, No. 6, 2021, pp. 31-37.
[15] RoshanJahan, PreetamSuman, Deepak Kumar Singh, An Algorithm To Secure Data For Cloud Storage, IT in Industry, Vol. 9, No.1, 2021, pp. 1382-1387.
[16] BrozolalMondol and Md. AshiqMahmood, An Efficient Approach for Multiple User Data Security in Cloud Computing, IEEE International Conference on Artificial Intelligence and Smart Systems, 2021, pp. 1130-1135.
[17] Md. Abu Musa and Md. AshiqMahmood, Client-side Cryptography Based Security for Cloud Computing System, IEEE International Conference on Artificial Intelligence and Smart Systems, 2021, pp. 594-600.
[18] Pronika and S. S. Tyagi, Secure Data Storage in Cloud using Encryption Algorithm,IEEE International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, 2021, pp. 136-141,
[19] Tahir, M., Sardaraz, M., Mehmood, Z. et al. CryptoGA: a cryptosystem based on genetic algorithm for cloud data security. Springer Cluster Computing, Volume 24, 2021, pp. 739–752.
[20] Adee, R.; Mouratidis, H. A, Dynamic Four-Step Data SecurityModel for Data in Cloud ComputingBased on Cryptography and Steganography, Sensors, 22, 1109, 2022, pp. 1-23.

A. Fairosebanu received her M.Sc., (CS) degree in Government Arts College (Autonomous), Kumbakonam, India, in 2012. She also received her M.Phil. (CS) degree in the Jamal Mohammed College (Autonomous), Trichy, India, in 2013. Now she is employed as an Assistant Professor in PG & Research Department of Computer Science, Idhaya College for Women, Kumbakonam, India. In addition, she is pursuing a PhD (Computer Science) in Government Arts and Science College, Kumulur, Lalgudi, Trichy, India.

Dr. A. Nisha Jebaseeli completed her PhD (Computer Science) at Bharathidasan University in 2014. Now she is employed as an Assistant Professor and Head, PG & Research Department of Computer Science, Government Arts and Science College, Kumulur, Lalgudi, Trichy, India. Now she is guiding 5 PhD Scholars. She has completed her M.Sc in Bishop Heber College, Trichy, India and M.Tech in Bharathidasan University, Trichy, India. She has 18 years of experience in Teaching and 5 years of experience in Research.

.