[1] B. Umapathy

[1*]G. Kalpana

User-Centric Cloud Framework for Enhanced Data Security by Applying UK Algorithm

JES Journal of Electrical Systems

Abstract: -Cloud storage is the technical boon which has become the realm of many companies like Amazon, Google, Microsoft, Drop box and many more. Remarkably all these companies are US based and thus eventually the personal data of every other nation is stacked on the Western companies. This leads the dependant country to a state of vulnerable. However if the dependant countries need to get autonomy they had to be provided with their own cloud storage which is impossible with the current mechanisms. With the facility of high computational devices and wide infrastructure and with the aid of secured encryption the cloud storage providers strongly protect the data of the users and enable their best service to the users. Also the cloud storage providers use the encryption algorithm in the provider infrastructure and data from user to provider infrastructure is done using the secured transmission. Finally the powerful infrastructure does the work to secure the data. This mechanism is certainly possible in the developed countries. But it is relatively impossible in the developing countries like India as it cannot afford the huge infrastructure to protect the public data. Even done, it might not result in generating profit because the subscription rate is comparatively lower in India than in developed countries. So to pursue the less profitable module is not suggestive and hence the reliability on western country cloud storage is over whelming. However to overcome this set back and to implement indigenous cloud storage this paper has attempted to implement the UK algorithm in user device environment through which mini and micro level cloud storage shall be developed resulting in the minimal cost of maintaining the cloud storage. In this case, cloud Storage provider do not need to have either the huge infrastructure nor the computational devices to process the data and no need of separate data transmission to secure the data as well. Employing the encryption algorithm in user environment helps to give full authority to the user on their data. The CS provider holds only encrypted data of the user and with low cost consumption this method is feasible even for the developing country like India.

Keywords: Cryptography, Cloud storage, Encryption, Cloud computing.

I. INTRODUCTION

Primarily, to safeguard the security of cloud data some of the prerequisite components are; Availability, Confidentiality, Accountability and Integrity. To facilitate the Cloud data security strategy CS facility is always preferred because of its high repute and much sought after demand. This has paved way for advanced development, making it necessary to update cryptographic systems to match the demands of the modern-days [1]. Cryptography is a technique which is used to secure data and protect it from unauthorized access. In the context of cloud computing, cryptography can be used to secure data that is stored in the cloud or transmitted over the internet. Cryptography plays a major role in cloud data security by providing mechanisms to secure data both at rest and in transit. It is important to choose the right cryptography techniques and algorithms that meet the specific security requirements of the data and the cloud environment. Google Drive stores metadata about the files and folders stored in the service, including information such as file type, size, creation date, and modification date. The privacy implications of this metadata should be considered when using Google Drive or other cloud storage services.

Permanent advancement in the field of data security has been made since the late 1990s under the direction of the National Institute of Standards and Technology (NIST). The encryption algorithms DES, AES, and RSA were also created at the same time. The AES (Rijndael) was given NIST approval in 2001 as a reliable safe for storing documents in encrypted form. Because of the AES's outstanding performance, the U.S. government, the military, and the FBI were subsequently able to employ it due to its cross-platform compatibility, efficient RAM and ROM management, quicker key setup, key expansion approach, and block cipher method. Since its introduction in 2001, the AES algorithm has been one of the most popular cryptographic algorithms ever used [2].

[1,1*]Department of Computer Science, Faculty of Science and Humanities, SRM Institute of Science and Technology, Kattankulathur, 603203

Email: [1]ub225@srmist.edu.in, [1*]kalpanag@srmist.edu.in.

It should be noted that the four types of cloud environment models—Private Cloud, Public Cloud, Community Cloud, and Hybrid Cloud—are broadly categorized based on the customers' access privileges offered by CSPs[3]. Since a few years ago, it has been noted that in spite of many cloud environment models; a significant portion of data is sensitive, necessitating the highest level of data protection. Although CSPs (Cloud Service Providers) guarantee data availability and integrity for large data storage in the cloud, data owners still worry about confidentiality, especially when data is transferred to a public cloud environment. A user-centric cloud framework for enhanced data security has thus become a must and this paper has initiated to make it possible by applying UK Algorithm.

II. LITERATURE REVIEW

Maryam Kamal et al[4], the proposed frame work uses Mass Distributed Storage (MDS), one of the primary methods distributed cloud storage uses to store massive data; hence it is anticipated to operate more effectively on cloud storage than standard encryption methods. The suggested approach is a clever and effective one for preserving distributed cloud storage secrecy. It may be used as a filter to protect the privacy of rising amounts of data being saved in cloud storage. When used as a filter, the framework will use less computing power and much less memory.

Yogesh Gupta [5] has introduced two efficient distributed load-balancing techniques. The suggested load balancing algorithms are created and put into practice to accomplish global balancing in the cloud environment by properly redistributing the additional load to the set of neighbors of the overloaded server to eliminate local server queue instability. To prevent load unbalancing, the recommended ways equitably divide the workload among the storage servers. The first suggested technique is CDLB (Capability based distributed load balancing), which evenly distributes the workload across the storage servers using the server queue and service rate. The CDLB algorithm chooses the server with the highest available queue and service rate. By employing the server queue, service rate, and service time for each client request, the second suggested method, known as DDLB (Deadline based distributed load balancing), distributes the workload among storage servers. The DDLB algorithm chooses the server that can fulfill client requests in the allotted amount of time through a regression study of the performance of the suggested methods in terms of the number of client requests that were completed, the number of requests that were delayed, the total system response time, and the average server usage.

Pratima Sharma et al[6], The suggested architecture uses a bilinear pairing key generation process in conjunction with a cipher text attribute-based encryption technique to offer key generation procedures. A distributed system is used in the key generation process to create a more secure environment and prevent total reliance on a single authority. Additionally, it offers an access control method that guarantees that the registered user, with the assistance of data owner nodes, may only access cloud data. Using the Merkle root principle, the suggested block chain topology preserves the accuracy of the cloud data. The Blowfish, RSA, and SHA-2 algorithms are combined to form a novel hybrid cryptographic algorithm. The combination of symmetric and asymmetric algorithms provides the efficiency of the suggested system. Using the SHA-2 algorithm, the recommended solution offers exceptional security for data transfer over the internet [7].

Shaopeng Guan et al[8], proposed the combination of HDFS federation and HDFS high-availability technique to overcome fault tolerance and metadata failure. The architecture achieves safe and efficient ample data storage for securing the data blocks. Security of cloud data is a top priority since the cloud service provider is unreliable. The researchers' exploration of the vital cloud data challenges focuses on the processing and security difficulties. Many innovative approaches and technologies have been proposed to address security issues; block chain-based cloud technology is the most popular [9].

Suggested lightweight cryptographic protocols and multi-factor authentication is based on IOT for cloud storage systems. Sensitive and non-confidential data for IOT devices is categorized as follows: There are two aspects to confidential information. To ensure high security, data is encrypted using a different encryption technique (RC6, Feistel), and is stored in a private cloud memory. One algorithm is used to encrypt the less sensitive data (AES). It is kept in the open cloud. A trusted authority guarantees multi-factor certification, and user IP, passwords, and biometrics are supported for user identification [10].

Barman et al. [11] proposed a novel idea in which two communicating users would share a session key. Their biometric data is used to construct the cryptographic architecture. In a cryptographic framework, the biometric data of the two communicating parties is combined to produce a mutual locker. Users can lock their biometric data with each other using a mutual locker, which protects biometric data against interception while distributing lockers across a public network. Additionally, a mutual locker protects the counterparty's biometric data from an insider attacker who communicates with them legally using their personal key.

Krishnaveni et al. [12] suggested an intrusion detection system with a mix of chosen characteristics to counter honeypot assaults and enhance outcomes based on the Kyoto dataset. High-security data uploading methods employing elliptic curve keys for key generation and the blowfish algorithm for data encryption have been proposed by Mudepalli et al. [13]. This method is quite effective for both the owner and the user of the data. By utilizing replacement boxes, Haq et al. [14] updated the blowfish method for the digital picture to add more mathematical complexity. Suhasini and Kanchana [15] have introduced a hybrid chaotic map which is utilized for encrypting the medical image during transmission in the cloud. Initially, the Arnold map is used for scrambling the initial value, and the tent map is used iteratively to define the values to locate the point of the plaintext pixel, the fractional Lorenz system takes the **moulded** pixel as the input, and scrambling is attained using a matrix method to achieve confusion. Additionally, it utilizes pseudo-random sequences to execute the cross-diffusion process to attain an encrypted medical image.

Gracy and Rebecca [16] [19] have proposed solutions to improve the efficiency of the block chain network, such as authentication, monitoring selection for an epoch, and mass mining using a modified proof of work algorithm. These solutions aim to address the challenges of privacy leakage and selfish mining and improve the overall performance of the block chain in terms of throughput and latency. Eventually presents a comprehensive view of block chain technology, security risks, and challenges that arise as a result of its sensitivity as a popular technology. Umapathy and kalpana [17][18] Proposed Novel UK Algorithm to increase the block complexity to avoid the brute force technique by high computational power computer in future. The algorithm helps to cloud service providers avoid maintaining the metadata of the data, and it helps the user data to high security. It is applicable to multiple file types. The key expansion is a unique technique to increase the complexity of the prediction. Further direction to implement the UK algorithm in a cloud environment is in progress.

III. PROPOSED METHOD

In this paper implementing the UK Algorithm in the user environment is proposed to decrease the maintenance cost of CS and increase the authority on data to the user. The Algorithm implemented is based on the client server architecture model. The client module holds the encryption algorithm to execute the encryption and decryption process of the data. The Server Module is responsible only for the data transfer to the client as per requesting the data.

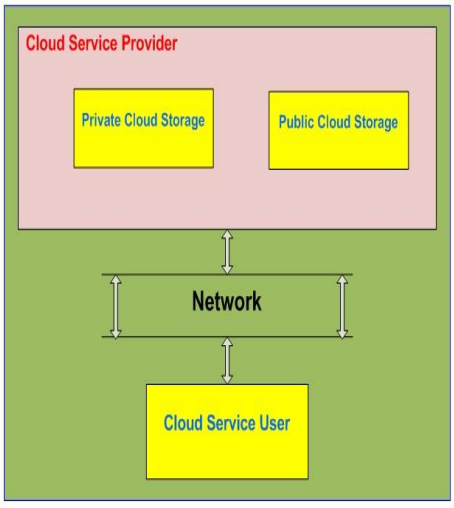

Figure 1. Cloud Storage Framework Model.

The proposed framework for a cloud storage management system has a cloud service provider (CSP) as the server module, Cloud Service User (CSU) as the Client module, and a network layer for communication purposes. The CSP is divided into two public and private cloud storages as a requirement of the UK Algorithm. The CSU configurations depend on the user's devices. In this architecture the user can determinate the usage and according to the usage requirement user can engage the system efficiency. Both the encryption and decryption process is executed in user end environment. The UK Algorithm do not store any metadata and password in the CS, it seems CS provider does not need the high level protection to secure the password and metadata. The encryption keys are managed only by the user end environment so the possibilities of the intruder to tamper the key are very less. The data owner becomes sole responsible for the key and UK algorithm provides the high level complexity to overcome the cryptanalysis technique.

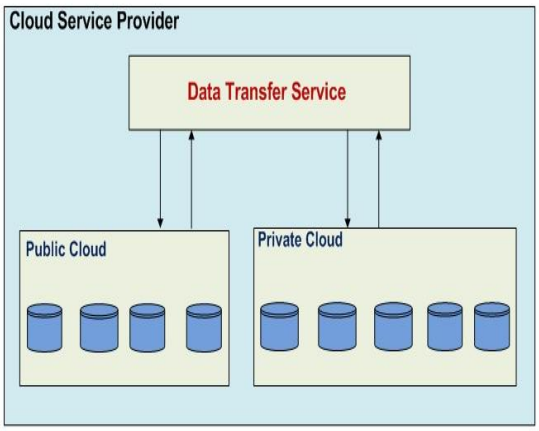

Figure 2. Cloud Service Provider Architecture

The Fig. 2 states that CSP architecture holds only one service (data transfer service), which is responsible for data transfer bidirectional between CSP and CSU. The architecture has two kinds of storage, public and private. The public storage holds the collective of all CSU's data. The private storage has separate space allocation for each CSU, and the allocated space is managed only by the particular user.

The Fig. 3 portrays CSU architecture holds five services: Data transfer services, Index services, Block processing services, Naming services, and Encryption decryption services. The data transfer services work as in CSP. The Index services display the user's data in private storage. Block processing services convert user data into binary data blocks. A naming service has worked to set a name for the blocks. The Encryption Decryption services are used to encrypt or decrypt the binary data blocks.

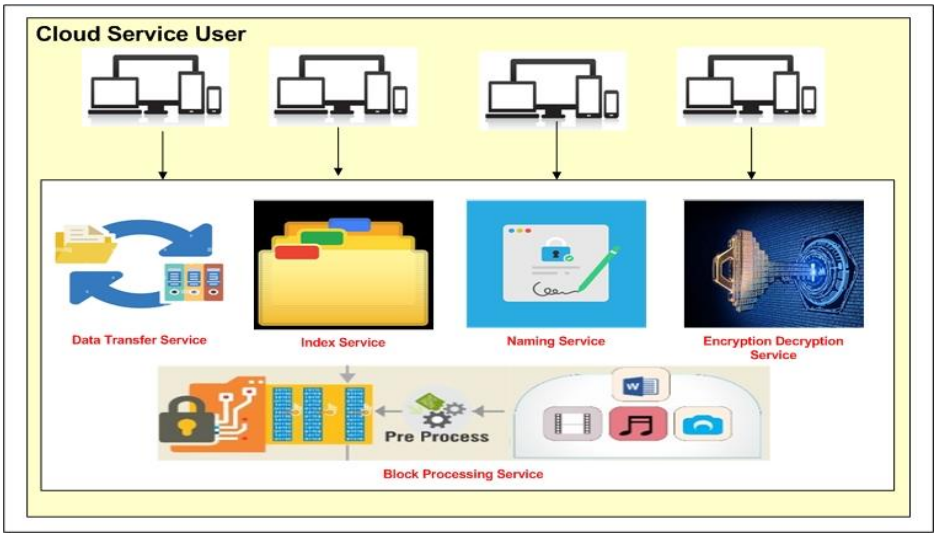

Figure 3. Cloud User Provider Architecture.

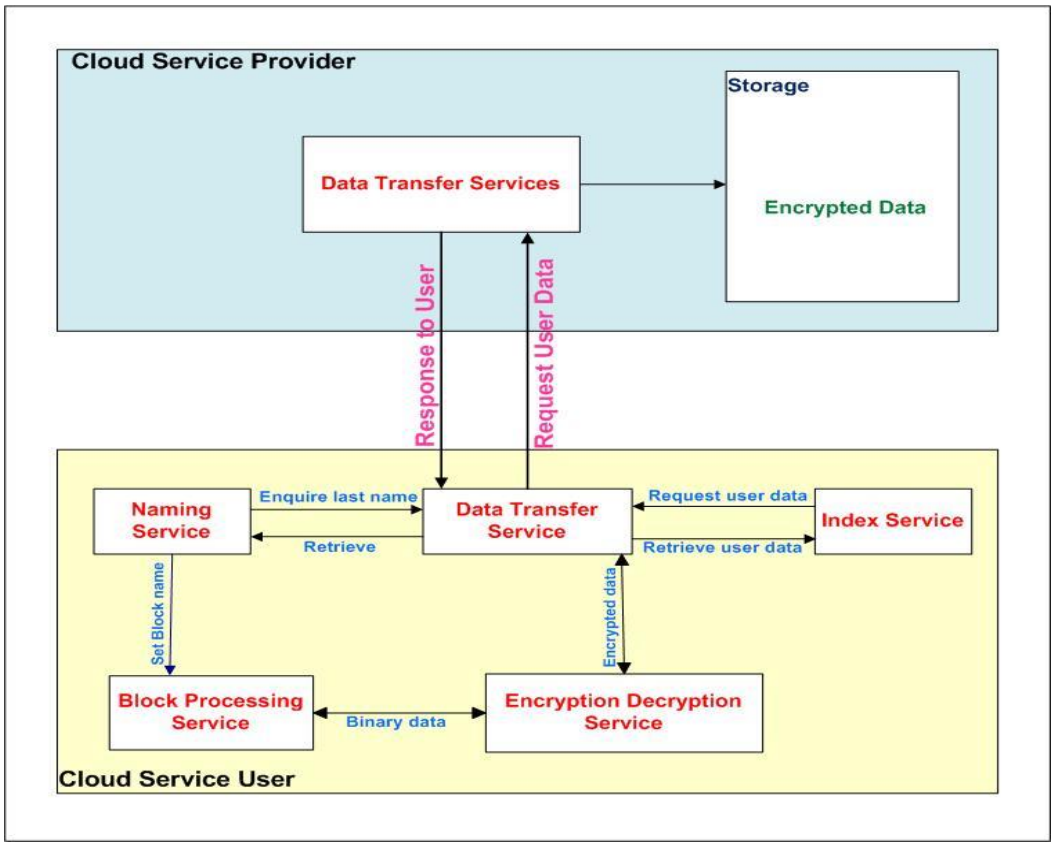

Figure 4. Service Flow Architecture

The Fig. 4 states service architecture helps to show the services' flow in the working environment, both CSP and CSU. The index services request the data from CSP private storage to view the individual user data through the data transfer service CSU to CSP. CSP sends the response to CSU; the response holds the user data. The naming services request the last file named in the CSP public storage and set the name to the new block according to the requested last file name. Block processing services are significant processes that read the user's original file and convert the original file into a binary block. The number of blocks depends upon the user's file size. The binary block sends to the encryption-decryption service. The block processing services do the reverse conversion process to the binary block received from the encryption-decryption service, resulting in original data to the user. The encryption-decryption block is responsible for encrypting and decrypting the binary block to secure data. The encrypted data is sent to CSP through data transfer services.

The Server architecture needed low computational power to perform the data transfer. In such case the low configured system can manage the huge volume of users efficiently. The comparison of the existing Server architecture and proposed Server architecture shows the efficiency is improved by minimizing the work and the cost of running the server is also minimal. The proposed Server architecture is suitable for personal, micro, mini levels of CS provider. In large scale scenarios CS provider can handle the multiple users easily than traditional architecture, in terms of huge volume of user at the time. It is probably twice or thrice than the traditional.

The server holds the public database and private database as per requirement of the UK algorithm. The private database is creating according to number of user, while registering the services. The private database holds the first encrypted block of the file and named as file name uploaded. The public database holds the second to last block of different users. The private database is indexed in user view of client module.

The UK algorithm is working on different file format so that the CS only hold the encrypted file of different file format and the encryption and decryption is executed in user environment. The data transmission holds encrypted files only on both uploading and downloading sessions. The complexity of the UK algorithm feature itself, gives the security form the intruder in case the intruder listens the transmission and gets the encrypted blocks. The keys

are not stored in CS and not transferred through transmission path. Even if the intruder holds the blocks, the complexity of algorithm protects the reading of data.

The indexer service helps to view the user account private storage data. The private storage maintains first encrypted block data for user identification. The block processing service read the file and converts it into binary data. After completing the conversion data is split to block form. Finally the encryption service holds the UK algorithm to encrypt the data and the client side data transfer services communicate to server data transfer services to store the data in cloud.

IV. RESULT AND DISCUSSION

To analyze the implementation of UK algorithm by implementation on scenario-based concepts, the experimental execution is done in python language and the system configuration as client is Intel Core i7 (2.60 GHz) with 16 GB of RAM, Intel Core i5 (3.20 GHz) with 8 GB of RAM and Raspberry Pi 4(Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz) with 8 GB of RAM. The server is Intel Core i7. The communication established through LAN network cable.

Table 1. Encryption time results of UK algorithm.

SL NO	FILE TYPE	SUB TYPE	SIZE IN BYTES	ENCRYPTION TIME - i7 (seconds)	ENCRYPTION TIME - i5 (seconds)	ENCRYPTION TIME – Pi 4 (seconds)
1	Document	xls	20,480	0.289	0.312	1.02
2	Document	doc	2,04,288	2.19	2.3451	7.65
3	Document	doc	10,28,096	12.9	14.911	46.32
4	Document	pdf	52,22,400	72.69	132.153	244.51
5	Image	jpg	24,576	0.257	0.407	1
6	Image	png	2,12,992	2.25	3.058	9.02
7	Image	jpg	10,44,480	13.7	15.775	47.61
8	Image	png	52,65,350	73.546	104.409	251.35
9	Audio	mp3	20,480	0.293	0.377	1.1
10	Audio	mp3	2,08,896	2.36	2.998	8.92
11	Audio	wav	10,73,152	13.43	15.445	47.92
12	Audio	mp3	52,89,384	74.056	80.191	237.9
13	Video	mp4	21,450	0.246	0.356	1.01
14	Video	mp4	2,11,796	2.29	2.789	8.68
15	Video	mp4	11,50,976	13.6	17.504	49.34
16	Video	mp4	52,53,880	73.489	82.456	233.09

Table.1 represents the encryption time of the UK Algorithm applying a similar dataset in different user environments. The file type and subtype show that UK Algorithm suits different file formats. The size in bytes represents that each file type has four size categories around 20 kb, 200 kb, 1 Mb, and 5 Mb. SL no 1,5,9,13 are similar size data in different format states irrespective of file format. The encryption time depends upon the file size. The encryption time of varying user environments in the UK Algorithm shows performance depends on the CSU and is not fully reliable on CSP.

Table. 2 represents the decryption time of the UK Algorithm applying a similar dataset in different user environments. The decryption time comparison of small-size files is Pi 4 processor performs slightly better than other environments. On the other hand, the file size increases, and the high-performing process is good. It shows the slight file size performance sound in the low-configured environment and the big file size performance sound in the high-configured environment.

Table 2. Decryption time results of UK algorithm

SL NO	FILE TYPE	SUB TYPE	SIZE IN BYTES	DECRYPTION TIME - i7 (seconds)	DECRYPTION TIME - i5 (seconds)	DECRYPTION TIME – Pi 4 (seconds)
1	Document	xls	20,480	3.89	7.116	5.07
2	Document	doc	2,04,288	7.95	10.37	12.77
3	Document	doc	10,28,096	19.78	22.48	62.86
4	Document	pdf	52,22,400	84.98	90.76	347.65
5	Image	jpg	24,576	4.09	7.784	4.09
6	Image	png	2,12,992	7.53	10.067	14.67
7	Image	jpg	10,44,480	20.61	23.884	63.22
8	Image	png	52,65,350	86.529	107.696	336.86
9	Audio	mp3	20,480	4.11	8.5	4.82
10	Audio	mp3	2,08,896	8.1	9.843	14.43
11	Audio	wav	10,73,152	20.89	23.447	64.54
12	Audio	mp3	52,89,384	85.956	95.015	387.49
13	Video	mp4	21,450	4.01	7.983	4.92
14	Video	mp4	2,11,796	7.53	10.27	13.88
15	Video	mp4	11,50,976	21.01	25.257	67.95
16	Video	mp4	52,53,880	86.365	93.957	319.72

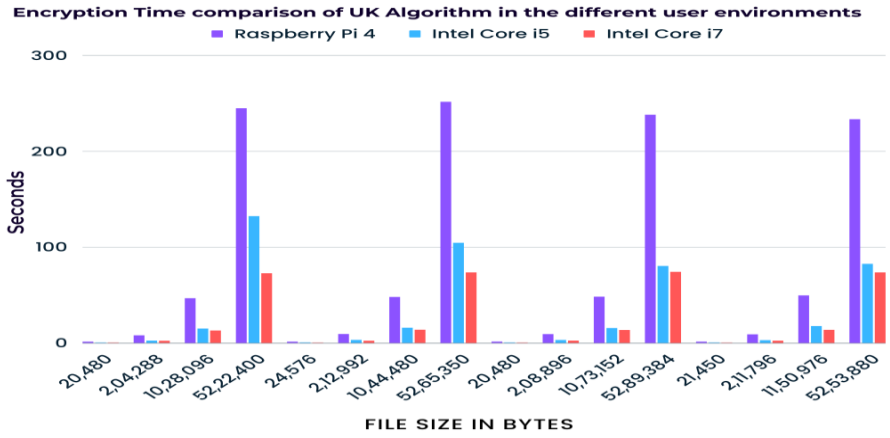

Figure 5. Encryption time comparison of the UK Algorithm in the different user environments

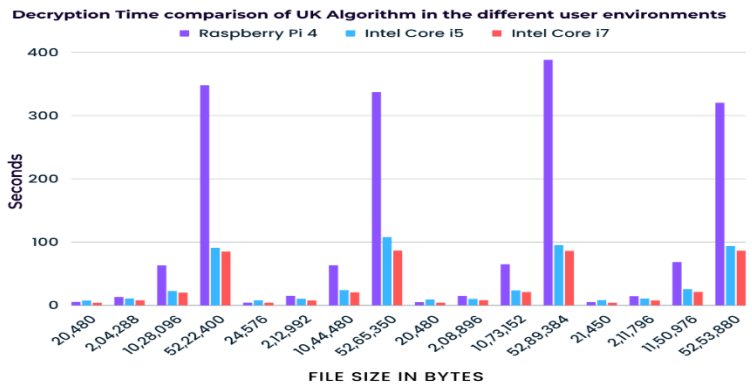

Figure 6. Decryption time comparison of the UK Algorithm in the different user environments

Fig. 5 & 6 and Table 1 & 2 represent the comparitive performance of the Encryption time and the Decryption time of the UK Algorithm in different user environments. It shows that the same file executed in three different configured user environments, resulting in different times based on executive power. The encryption time can also impact other aspects of system performance, such as CPU usage, memory usage, and storage requirements. The UK Algorithm performance increases when it executes in a high-powered user environment, and the performance decreases based on the high-to-low configuration. The Decryption time is higher than the Encryption time, and it is usually in the cryptographic algorithms to perform the order of the data to decrypt.

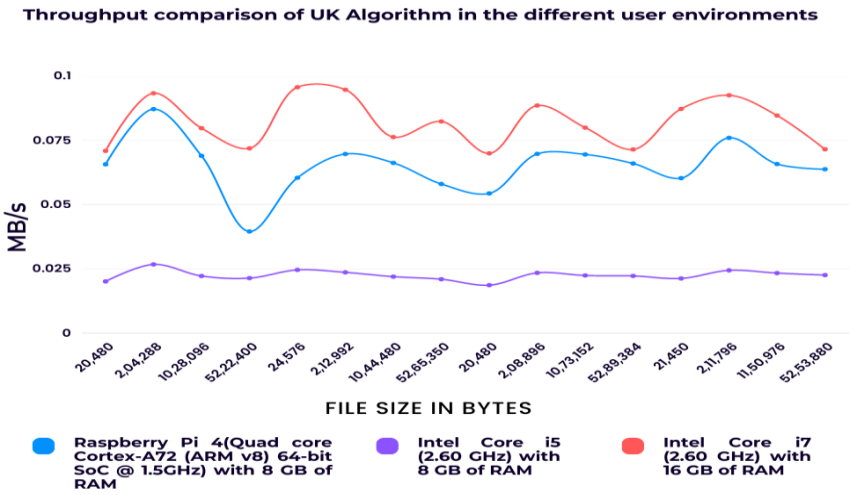

Figure 7. Throughput comparison of the UK Algorithm in the different user environments

Throughput (MB/s) = (Data Size in bytes) / (Time taken in seconds * 1,000,000) (1)

Average Throughput Time = Total Time / Number of Processes (2)

Fig.7 represents the Throughput comparison of the UK Algorithm in different user environments based on eq. 1. It shows the Throughput is high when the algorithm is executed in Intel Core i7 and averages around 0.075 MB/s. The small file size is performing below the average in Intel Core i7. The Intel Core i5 is middling between the i7 and Raspberry pi 4. The Throughput is low when the algorithm is executed in Raspberry pi 4 and averages around 0.025 MB/s. It states again the UK Algorithm performance of the proposed cloud storage architecture is subject to the user's environment.

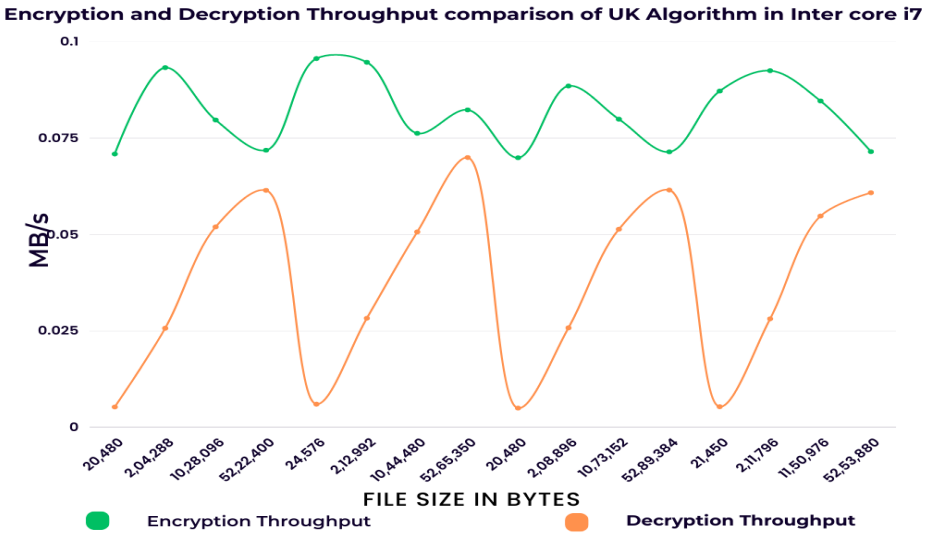

Figure 8. Encryption and Decryption Throughput comparison of UK Algorithm in Inter core i7

Fig.8 represents encryption and decryption throughput comparison of the UK Algorithm in Intel Core i7. It states the difference in execution speed of the small-sized file (20480) is lower. The contrast of execution speed is higher constantly when the file size increases. The big file size (5253880) increases the execution speed of the decryption Throughput and minimizes the gap in execution speed between encryption and decryption throughput

Table 3. Average Throughput time results of UK algorithm

User Environment	Average Throughput time for Encryption (Mb/s)	Average Throughput time for Decryption (Mb/s)
Intel Core i7	0.081880117	0.037006307
Intel Core i5	0.065034811	0.031248747
Raspberry Pi 4	0.022473981	0.012880481

Table .3 represents the Average Throughput time for Encryption and Decryption of the UK Algorithm in the different user environments based on eq. 2. The value is evaluated from equation 1, and the dataset is considered as the 16 files in table 1. It states that the CSU uses a highly configured system to access the cloud storage, and the speed of the execution is high compared to a mid or low-configured system. The average Throughput time for decryption is lower than encryption and applies to all user environments.

V. CONCLUSION

The implementation of UK algorithm in the client server architecture is very reliable model for the data security locker scenario. The proposed architecture is very useful to construct mini and micro level personal data storage. The Government document stored in this model gives more authority to the officials for the credibility. This scenario shows the data stored in the government database by the officials cannot be accessed by the other government officials including higher officials. In another scenario the blocked officials cannot access their credentials without the knowledge of the government. In future the model has to be tested in larger multi user dataset and the algorithm shall be upgraded based on continuous monitoring of the model.

CONFLICT OF INTEREST

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

AUTHOR CONTRIBUTIONS

This research was conceptualised by umapathy.B who also carried out the experiments, analysed the data, and wrote the manuscript. Dr. Kalpana.G supervised all processing and critically revised the manuscript for important intellectual content

FUNDING

The authors received no specific funding for this study.

REFERENCES

[1] M. Y. Pandith, “Data Security and Privacy Concerns in Cloud Computing,” Internet Things Cloud Comput., vol. 2, no. 2, p. 6, 2014, doi: 10.11648/j.iotcc.20140202.11.

[2] M. E. Smid, “Standard,” vol. 126, no. 126024, pp. 1–18, 2022.

[3] M. Amoon and A. A. Nasr, “A Management System for Servicing Multi-Organizations on Community Cloud Model in Secure Cloud Environment,” 2019.

[4] M. Kamal et al., “Microprocessors and Microsystems Privacy-aware genetic algorithm based data security framework for distributed cloud storage,” Microprocess. Microsyst., vol. 94, no. October 2021, p. 104673, 2022, doi: 10.1016/j.micpro.2022.104673.

[5] Y. Gupta, “Novel distributed load balancing algorithms in cloud storage,” Expert Syst. Appl., vol. 186, no. March, p. 115713, 2021, doi: 10.1016/j.eswa.2021.115713.

[6] P. Sharma, R. Jindal, and M. Dutta, “Journal of Information Security and Applications Blockchain-based decentralized architecture for cloud storage system,” J. Inf. Secur. Appl., vol. 62, no. September, p. 102970, 2021, doi: 10.1016/j.jisa.2021.102970.

[7] D. P. Timothy, “A Hybrid Cryptography Algorithm for Cloud Computing Security,” vol. 5.

[8] “6.Hadoop-based secure storage solution for big data in cloud computing environment _ Elsevier Enhanced Reader.pdf.” .C. Xu, K. U. N. Wang, M. Guo, C. Xu, and T. K. Wang, “INTELLIGENCE IN THE CLOUD Intelligent Resource Management in Blockchain-Based Cloud Datacenters,” pp. 50–59.

[9] S. Atiewi et al., “Scalable and Secure Big Data IoT System Based on Multifactor Authentication and Lightweight Cryptography,” vol. 8, 2020, doi: 10.1109/ACCESS.2020.3002815.

[10] S. Barman, S. Chattopadhyay, D. Samanta, and G. Panchal, “A novel secure key-exchange protocol using biometrics of the sender and receiver,” Comput. Electr. Eng., vol. 64, pp. 65–82, Nov. 2017, doi: 10.1016/J.COMPELECENG.2016.11.017.

[11] S. Krishnaveni, S. Prabhakaran, S. Sivamohan, and S. Sridhar, “Network intrusion detection based on ensemble classification and feature selection method for cloud computing,” no. November 2021, pp. 1–29, 2022, doi: 10.1002/cpe.6838.

[12] S. Mudepalli, V. Srinivasa Rao, and R. Kiran Kumar, “An efficient data retrieval approach using blowfish encryption on cloud ciphertext retrieval in cloud computing,” Proc. 2017 Int. Conf. Intell. Comput. Control Syst. ICICCS 2017, vol. 2018-Janua, pp. 267–271, 2017, doi: 10.1109/ICCONS.2017.8250724.

[13] T. U. Haq, T. Shah, G. F. Siddiqui, M. Z. Iqbal, I. A. Hameed, and H. Jamil, “Improved Twofish Algorithm: A Digital Image Enciphering Application,” IEEE Access, vol. 9, pp. 76518–76530, 2021, doi: 10.1109/ACCESS.2021.3081792.

[14] P. Suhasini, “Enhanced Fractional Order Lorenz System for Medical Image Encryption in Cloud-Based Healthcare Administration,” vol. 9, no. 4, pp. 424–437, 2022, doi: 10.22247/ijcna/2022/214504.

[15] M. Gracy, “A SYSTEMATIC REVIEW OF BLOCKCHAIN- BASED SYSTEM : TRANSACTION THROUGHPUT LATENCY AND CHALLENGES Department of Computer Science,” 2021.

[16] B. Umapathy and G. Kalpana, “A novel symmetric cryptographic method to design block complexity for data security,” Comput. Electr. Eng., vol. 104, no. PB, p. 108467, 2022, doi: 10.1016/j.compeleceng.2022.108467.

[17] B. Umapathy and G. Kalpana, "A Key Generation Algorithm for Cryptographic Algorithms to Improve Key Complexity and Efficiency," 2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2023, pp. 647-652, doi: 10.1109/ICSSIT55814.2023.10060906.

[18] B. Rebecca Jeyavadhanam, M. Gracy, V. V. Ramalingam, Christopher Xavier, “Fundamental Concepts and Applications of Blockchain Technology”, Big Data Analysis for Green Computing, Taylor and Francis, 2021,DOI https://doi.org/10.1201/9781003032328]

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 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.