Bulletin of Electrical Engineering and Informatics
Vol. 15, No. 4, August 2026, pp. 3135~3145
ISSN: 2302-9285, DOI: 10.11591/eei.v15i4.10756

Security concern based Damgard–Jurik algorithm for data transmission in decentralized cloud storage

Karuppasamy Lakshmanan, Vasudevan Venkatraman
Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education, Srivilliputhur, India

Article Info

Article history:

Received May 27, 2025
Revised Apr 10, 2026
Accepted Jun 18, 2026

Keywords:

Damgard–jurik algorithm
Decentralized cloud storage
Dual chunk redundancy
Encryption
Slicing

ABSTRACT

Decentralized cloud storage (DCS) provides IT resources to a growing community of users. Decentralized storage systems offer availability, redundancy, and security because the data will be spread across numerous nodes. Despite its benefits, there are still challenges associated with DCS, such as inconsistency in data versions, complicated data retrieval, and the problem of data integrity and privacy. To effectively handle these challenges, it is necessary to come up with new solutions. To overcome these issues, a novel Dual-Chunk redundancy assisted Damgard–Jurik-incorporated slice-based data security (DRAGO-SLICE) framework has been suggested in this paper to improve security and reliability in DCS. The suggested approach utilizes the Damgard–Jurik algorithm (DJA) for encrypting the cloud resource data. The slicing process divides encrypted data into multiple chunks to increase security. Enhanced system reliability has been achieved by implementing dual chunk redundancy (DCR). Python has been used to simulate the suggested model. The efficacy of the developed approach is evaluated utilizing metrics namely decryption time (DT), security strength, encryption time (ET), latency, computational overhead, reliability, storage efficiency, and throughput. The proposed DRAGO-SLICE strategy performs better in terms of security than the existing methods, including Ethereum virtual machine elliptic-curve cryptography (EVM-ECC), decentralized blockchain-based security (DeBlock-Sec), and blockchain-based decentralized storage system (BC-DSS) approaches, by 21.05%, 13.74%, and 8.52%, respectively.

Corresponding Author:

Karuppasamy Lakshmanan
Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education
Srivilliputhur, India
Email: karuppasamy40@outlook.com

1. INTRODUCTION

Decentralized cloud storage (DCS) is an emerging phenomenon in data management [1]. DCS provides security, fault tolerance, privacy, and scalability compared to the centralized cloud storage (CCS). CCS data is centrally stored on the server, which is controlled by one authority [2], [3]. In DCS, it transmits the data to several nodes on a peer-to-peer (P2P) network [4]. The architecture reduces uncertainty related to a single point of failure, data breaches, and service outages. Nonetheless, it is severely challenged with security issues such as data integrity, confidentiality, and access control (AC) [5]-[8].

Encryption has become quite relevant in the concerned security issues as it protects data against unauthorized access and maintains its confidentiality [9]. With the use of sophisticated encryption method, information stored on distributed nodes cannot be compromised in case some nodes are damaged [10]. Moreover, encryption improves integrity of data by ensuring that it is not modified by unauthorized third

parties, as well as determining the authenticity of data. Despite its potential, DCS still faces numerous challenges, particularly in balancing security, efficiency, and accessibility [11]-[13]. Issues such as key management, encryption overhead, and trust among decentralized nodes can hinder system performance [14]. Additionally, the distributed nature of these systems can introduce complexities in data retrieval and consistency [15]. One of the most important issues in DCS research is making sure that encrypted data is only accessible by authorized users without sacrificing performance [16].

The existing ways of DCS still bear several research gaps, including high latency, greater computational cost, inefficient key-management plans, and diminished scaling as the network grows [17]. The majority of the classical encryption methods fail to facilitate effective processing at the distributed nodes and cannot effectively coordinate with the slicing or redundancy mechanisms. Furthermore, the use of multifaceted cryptographic algorithms can tend to consume more resources, and real-time access to data can be difficult [18]. To overcome these problems, a novel dual-chunk redundancy assisted Damgard–Jurik-incorporated slice-based data security (DRAGO-SLICE) framework has been suggested to enhance security and reliability in DCS. The main contributions of the DRAGO-SLICE technique have been given as follows:

- The goal of the work is to improve the privacy, security, and reliability of the system in DCS.
- The suggested approach achieves secure, decentralized, and reliable data through the use of Damgard–Jurik algorithm (DJA), data slicing, and dual chunk redundancy (DCR) techniques.
- The proposed system utilizes the DJA for encryption to enhance data confidentiality and secure processing.
- The effectiveness of the proposed DRAGO-SLICE scheme has been examined parameters including encryption time (ET), security strength, storage efficiency, computational overhead, latency, reliability, throughput, and decryption time (DT), respectively.

The rest of the work is provided as follows: section 2 examines the literature survey. The recommended DRAGO-SLICE technique is provided in section 3. Section 4 has the result and discussion, whereas section 5 examines the conclusion.

2. LITERATURE REVIEW

Over the past few years, various papers have examined various methods of achieving data security in DCS applications. The subsequent section observes these contemporary methods and points out their respective constraints.

Khan *et al.* [19] suggested an Ethereum virtual machine elliptic-curve cryptography (EVM-ECC) framework for efficient data storage across a DCS using the Ethereum blockchain. This suggested solution offered a distributed paradigm for a role-based cloud AC framework. This paradigm enhanced data security by encrypting and distributing data among several peers in the architecture.

Narayanan *et al.* [20] suggested a novel decentralized blockchain-based security (DeBlock-Sec) solution to address the security problem, which would be most appropriate in an internet of things (IoT) environment with minimal resources. The comprehensive evaluations conducted in the Spark environment demonstrate that the suggested research performs better in terms of throughput, storage space, and time consumption.

Liu *et al.* [21] suggested a blockchain-based cloud computing AC system where the access policy is described by an access matrix that is kept up to date. While the suggested method has a larger communication overhead than PpBAC and SVPAC by 39.05% and 5.88%, respectively, it is more secure and has a 17.16% lower communication overhead than Timely CP-ABE.

Mishra *et al.* [22] presented a secure decentralized public auditing system based on blockchain technology in a DCS system that uses an effective deduplication strategy. Furthermore, a complete concrete security analysis shows that the suggested paradigm is computationally impractical in the face of powerful attacks such as data privacy, duplicate faking (DFA), forgery, storage free-riding, proof-of-ownership, and collusion.

Shakor *et al.* [23] presented an innovative dynamic encryption technique designed to improve the security of file storage in cloud environments. This method makes use of a hybrid dynamic encryption system that combines aspects of blockchain, advanced encryption standard (AES), and ECC. All these factors combine to enhance consumer confidence and enhance the security of information stored in the cloud.

Li *et al.* [24] proposed a blockchain-based decentralized storage system (BC-DSS) with a stable deduplication and storage balance mechanism to ensure the privacy of outsourced data. The results of simulation showed that the proposed framework maintains the confidentiality of outsourced data at a reasonable computational cost.

Sharma *et al.* [25] introduced a new strategy, which utilizes the interplanetary file system (IPFS), a secure and reliable connection network that can be used by web browsers and enables global data distribution

and storage on a decentralized P2P network. The analysis offers a viable method that does not interfere with user privacy but maintains usability without increasing costs.

As per a literature survey, various data security measures in DCS have been proposed. Nevertheless, it has drawbacks such as complicated encryption algorithms, poor scalability and vulnerability to security attacks. In order to address this challenge, this paper has proposed a novel DRAGO-SLICE method that is discussed in the next section.

3. PROPOSED SYSTEM

In this section, novel DRAGO-SLICE framework has been proposed to augment security and reliability in DCS. The general block diagram of the proposed model is shown in Figure 1.

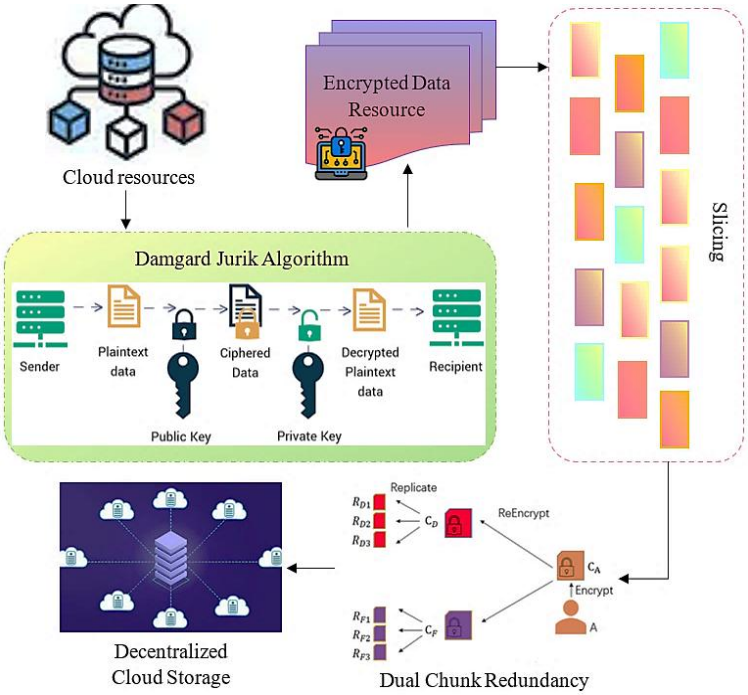


Figure 1. Proposed model

3.1. Encryption using Damgard–Jurik algorithm

The Damgard–Jurik encryption algorithm is a flexible cryptographic scheme, which aims to provide efficiency and security. It is based on the RSA cryptosystem with homomorphic properties, which enables one to safely perform calculations on encrypted data. It is a type of hybrid encryption that combines the characteristics of RSA and homomorphic encryption and is used to encrypt data before storing it on cloud servers.

Based on RSA architecture, the first step in this scheme is to generate the public and the private keys, which enable efficient exchange of keys and still retain their homomorphic properties. These keys are employed to transform every unit of data in a coded form upon encryption to ensure secrecy and integrity throughout the transmission and storage processes. The asymmetric Damgard–Jurik cryptosystem is defined as (1). Let $((r, MI_n), MI_x)$ be the public/private key pair.

$$MI_n = nm \; and \; MI_x = LCM((n-1),(m-1)) \quad (1)$$

Where n and m are two huge prime integers, and LCM stands for the least common multiple function. $\mathbb{Z}_{MI_n^a} = \{0,1,\ldots,MI_n^a - 1\}$ and $\mathbb{Z}^*_{MI_n^a}$ indicates the integers with modulo multiplicative inverses MI_n^a where m is an ordinary amount and choice $r \in \mathbb{Z}^*_{MI_n^2}$ which is given in (2):

$$r = (1 + MI_n)^u s \; mod MI_n^{a+1} \quad (2)$$

Where, u is a recognized mutable that moderately prime with MI_n and $s \in \mathbb{Z}^*_{MI_n^2}$. The Damgard–Jurik encryption of a plain-text $p \in \mathbb{Z}_{MI_n^a}$ into the cipher-text $y \in MI_n^{a+1}$ using the public key MI_n is given in (3):

$$y = Q[p, g] = r^p g^{MI_n^a} mod MI_n^{a+!} \quad (3)$$

Where, $g \in \mathbb{Z}_{MI_n^a}$ is the Damgard–Jurik encryption system stochastic or semantically secure, given a random number associated with p. More specifically, depending on the value of g, the encryption of the same plain-text message will produce different cipher-texts using the same encryption key. Selecting $r = 1 + MI_n$ will enable a speedy implementation of this cryptosystem without compromising algorithm security. There are two processes involved in decrypting y using the private key MI_x, based on the premise that $r = 1 + MI_n$. Initially, the decoder calculates in (4):

$$y^{MI_x} = (1 + MI_n)^{MI_x p} mod MI_n^{a+1} \quad (4)$$

To get access to the message p from y^{MI}, the decipherer has to compute $MI_x p$. To do so, Damgard–Jurik proposed a reiterative process to find p from $(1 + MI_n)^p mod MI_n^{a+1}$. The decoding of the cipher-text y into p is given in (5):

$$p = F(y^{MI_x}).MI_x^{-1} mod MI_n^a \quad (5)$$

By combining the strengths of RSA encryption with homomorphic properties, the DJA safeguards sensitive data within cloud environments, fortifying against unauthorized access or manipulation. The data deposited in the cloud server in the encrypted form is decrypted using the Damgard–Jurik method, and if the data user is authorized to deliver the search result, the search result is then safely sent to the data user.

The main defense is provided by the key-driven encryption, and security is further enhanced by the usage of DJA. A ciphertext is created from a plaintext resource using a DJA, which has the drawback that the entire transformation must be completed in order to recover the original plaintext. In actuality, DJA ensures that every bit of the encrypted resource is completely dependent on every other bit so that if any part of the encrypted resource is unavailable, no part of the original plaintext can be recovered.

3.2. Slicing

The process of slicing an encrypted resource involves creating distinct fragments, each of which contains a mini-block that ensures that no two fragments have the same mini-block, and introducing a fragment that contains each mini-block. We slice the resource by simply placing the mini-blocks that appear at the same location in the various macro-blocks in the same fragment in order to guarantee all of this and to make management easier. Figure 2 shows the process of slicing.

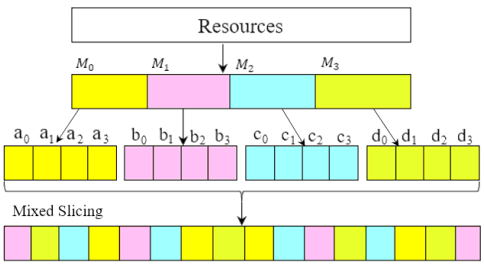


Figure 2. Slicing process

3.3. Dual chunk redundancy

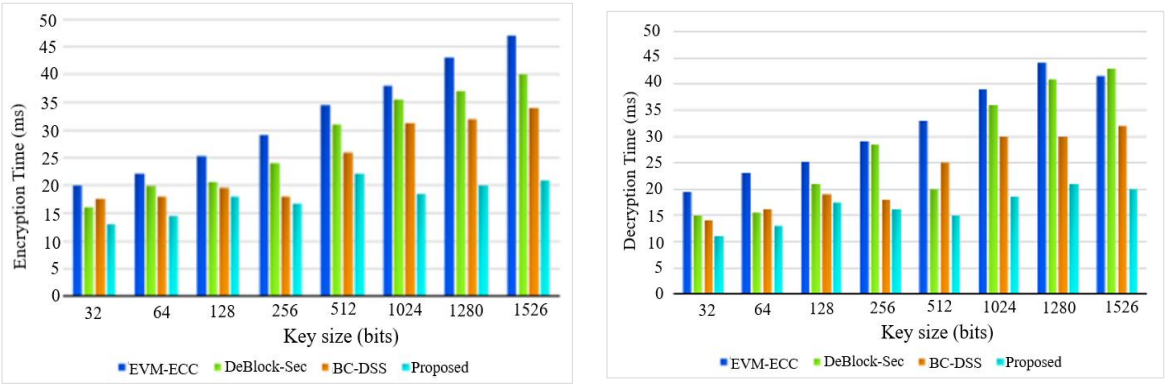
This study proposes a DCR-based DRAGO-SLICE technique, which consists of two key phases: trust management (TM) modules and DCR storage modules. TM phase involves the analysis of user behavior to calculate a trust value. During the dual modular redundancy storage stage, every piece of data is reproduced two times in independent storage nodes in order to provide fault tolerance and data uniformity. The concept of dual redundancy is based on a verification-based system, where the first copy is the main storage and the second one is a backup. To handle its data corruption or attack, the error detection mechanisms are applied to both chunks and in the event of discrepancy, the system applies the integrity verification system to determine the correct version.

```
Pseudocode for Draco-Slice framework:
    1.  (P_K,S_K) ← DJA_KeyGeneration() // DRACO_SLICE(Data)
    2.  EncData ← ∅
    3.  For each block B_i in Data do    //Encryption Phase
C_i ← DJA_Encrypt (B_i,P_K)
EncData ← EncData ∪ C_i
    4.  End for
    5.  Slices ← Slice_EncryptedData (EncData) //Slicing phase
    6.  RedundantChunks ← ∅      // Dual chunk redundancy
    7.  for each slice S_j in Slices do
RedundantChunks ← RedundantChunks ∪ {S_j, Duplicate (S_j)}
    8.  end for
    9.  for each chunk C_k in RedundantChunks do
Node ← Select_DCS_Node ()
Store_Chunk (C_k,Node)
    10. end for
    11. return "Data Stored Securely"
    12. if Authenticate (User) = FALSE then   // DRACO_Retrieve (DataID, User)
return "Access Denied"
    13. end if
    14. Chunks ← Fetch_Chunks (DataID)
    15. ValidChunks ← ∅ // Integrity Check using redundancy
    16. for each pair (C_1,C_2) in Chunks do
ValidChunks ← ValidChunks ∪ Integrity_Verify (C_1,C_2)
    17. end for
    18. PlainData  ← ∅ // Decryption Phase
    19. for each cipher C_i in ValidChunks do
M_i ← DJA_Decrypt (C_i,S_k)
PlainData ← PlainData ∪ M_i
    20. end for
    21. return PlainData
```

4. RESULT AND DISCUSSION

In this part, the proposed DRAGO-SLICE system is simulated in a Python environment. To investigate the effectiveness of the developed DRAGO-SLICE framework, it has been contrasted with the other methods, such as EVM-ECC [19], DeBlock-Sec [20], and BC-DSS [24]. A number of significant efficiency metrics including ET, DT, throughput, latency, computational overhead, and reliability were evaluated in order to regulate in what manner the DRAGO-SLICE method performed.

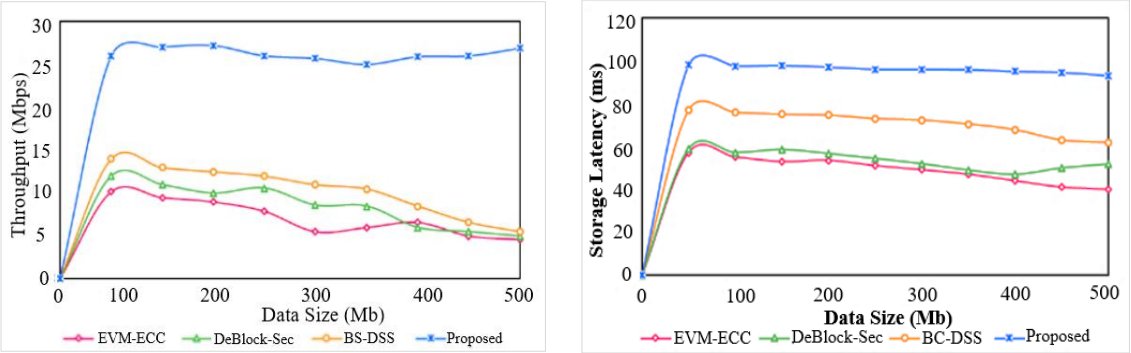
Figure 3 illustrates the proposed technique compared with existing approaches such as EVM-ECC, DeBlock-Sec, BC-DSS. Figure 3(a) shows a contrast between a proposed technique and existing methods like EVM-ECC [19], DeBlock-Sec [20], and BC-DSS [24] in terms of ET. The time booked to encode the data is quicker using the proposed approach when compared to other current solutions. The ETs of the proposed approach are 44.43%, 36.16%, and 26.63% quicker than those of the current techniques. Figure 3(b) compares the DTs of a suggested approach with the existing techniques. The DT is the amount of time required to utilize a decryption technique to return encrypted data to plaintext. The suggested method's DT is 48.07%, 40%, and 28.26% quicker than that of the existing methods.



(a) (b)

Figure 3. Comparison of: (a) ET and (b) DT

Figure 4 shows the throughput comparison of proposed technique with existing techniques, EVM-ECC, DeBlock-Sec and BC-DSS. Figure 4(a) presents a comparison of throughput in terms of data size. In this case, the data amount is continuously expanded, and throughput is indirectly proportional to the existing tasks. There are little variations in the graph of the throughput curve of the suggested DRAGO-SLICE. The existing systems have lower throughput. The proposed system shows that the involvement of the DJA enhances the throughput in the authentication process, encryption phase, and retrieval phase. The storage efficiency attained by the suggested study has been compared with earlier studies in Figure 4(b). According to the comparison, the suggested research achieves a storage efficiency of up to 98%, implying that 98% of the data is processed effectively and uninterrupted. Here the cloud is a decentralized environment where the authentication credentials are kept.



(a) (b)

Figure 4. Comparison of: (a) throughput and (b) storage efficiency

The Table 1 indicate that the suggested DRAGO-SLICE framework is more effective in the three resilience cases when compared to the existing approaches. In conditions of key compromise, it has the best accuracy and is more resistant to key related attacks. The suggested model ensures a better throughput and packet delivery ratio during node failure because of DCR and decentralized distribution. In the case of data losses, it has the maximum data-integrity value, that guarantees efficient recovery, and a low leakage of information.

Table 1. Resilience performance comparison across critical failure scenarios

Scenarios	Metrics	EVM-ECC (%)	DeBlock-Sec (%)	BC-DSS (%)	Proposed (%)
Key compromise	Accuracy	82	87	91	97
Node failure	Throughput	65	72	80	95
Network failure	PDR	73	78	87	94
Data loss	Data integrity	88	90	94	98

Figure 5 illustrates the computational overhead over time for four different methods: EVM-ECC, DeBlock-Sec, BC-DSS, and the proposed DRAGO-SLICE approach. It is evident that the suggested approach constantly exhibits the lowest computational overhead, starting from minimal values and increasing gradually to a stable level below 30 ms. Conversely, EVM-ECC has the highest overhead, surpassing 80 ms over time, whereas DeBlock-Sec and BC-DSS have an intermediate performance, with the overhead of around 60 ms and 40 ms, respectively.

As indicated in the Table 2, the proposed DRAGO-SLICE technique consumes much less additional storage compared to the current methods. Although the techniques such as EVM-ECC, DeBlock-Sec need high overhead and huge redundant copies, the overhead of the proposed technique is only 15 MB and the redundancy is 40 MB. This shows that DRAGO-SLICE provides high security and reliability at a significantly reduced storage usage in comparison to other procedures.

Proposed technique compared with EVM-ECC, DeBlock-Sec and BC-DSS in terms of latency, and data confidentiality in Figure 6. Figure 6(a) shows the suggested DRAGO-SLICE method, which has the lowest transmission latency. Traditional data redundancy mechanisms often introduce delays due to complex encryption and storage processes. In contrast, the DRAGO-SLICE model minimizes latency while maintaining high reliability through optimized chunk redundancy and adaptive data slicing. Therefore, the proposed approach is perfect for secure and resilient DCS systems. Figure 6(b) shows the data confidentiality rate as the number of the user data increases using four techniques such as EVM-ECC, DeBlock-Sec, BC-DSS, and the proposed method. The confidentiality of the proposed DRAGO-SLICE reaches its highest,

about 96 percent, when the size of the data increases to 100. BC-DSS and DeBlock-Sec are moderately enhanced; meanwhile, EVM-ECC exhibits the lowest levels of confidentiality.

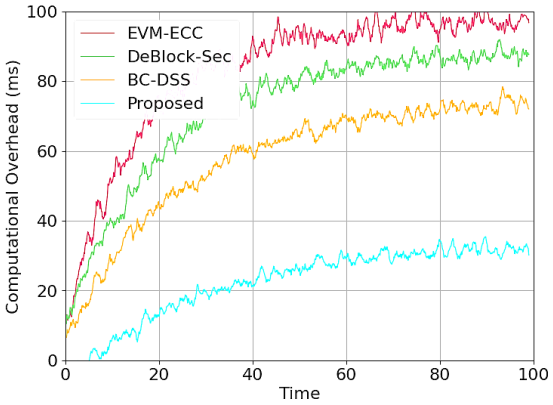
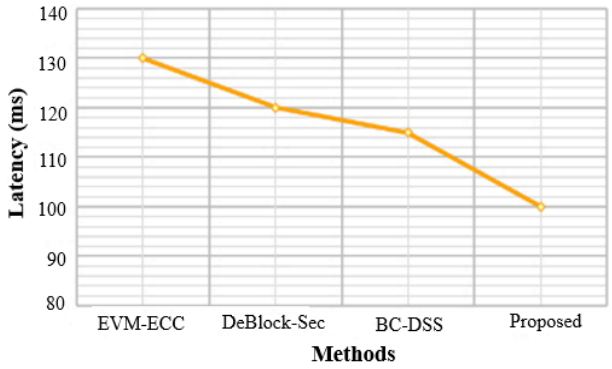


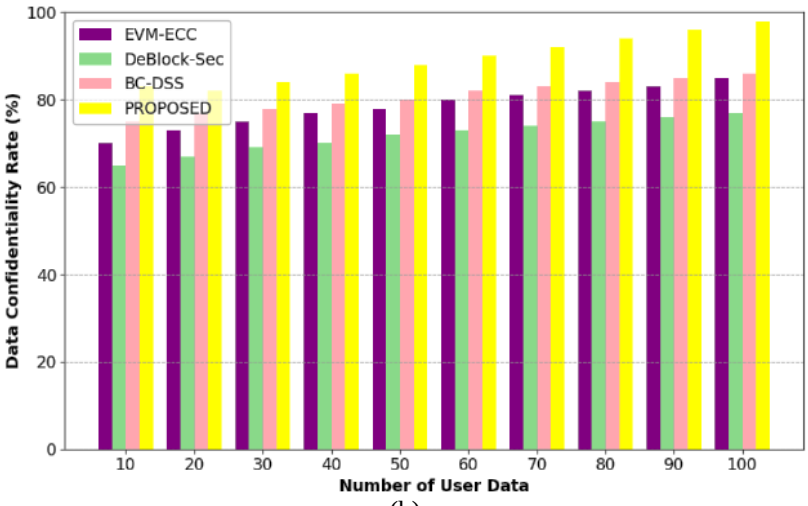
Figure 5. Computational overhead comparison

Table 2. Comparison of storage overhead and redundancy across security methods

Methods	Storage overhead (MB)	Redundancy (MB)
EVM-ECC	55	120
DeBlock-Sec	40	95
BC-DSS	28	70
Proposed	15	40



(a)



(b)

Figure 6. Performance evaluation of proposed method across different methods: (a) latency comparison and (b) data confidentiality

The Table 3 indicates the performance of EVM-ECC, DeBlock-Sec, BC-DSS, and the suggested DRAGO-SLICE method in seven metrics. The suggested scheme scores highest in every category, including encryption (87.23%), decryption (88%), throughput (95%), storage efficiency (98%), and the strength of security (97%). It is also the most reliable (99%) and the least latent (90.65%) which can be mostly attributed to the combination of Damgard–Jurik encryption, slicing, and dual-chunk redundancy.

Table 3. Comparative performance analysis of existing methods

Methods	Encryption (%)	Decryption (%)	Throughput (%)	Storage efficiency (%)	Security strength (%)	Reliability (%)	Latency (%)
EVM-ECC	55.7	51.93	65	75	78.95	70	65
DeBlock-Sec	63.84	60	72	80	86.26	78	72
BC-DSS	73.37	75	80	85	91.48	82	78
Proposed	87.23	88	95	98	97	99	90.65

Table 4 indicates that the proposed DRAGO-SLICE has the highest mean reliability (98.00) when compared with EVM-ECC, DeBlock-Sec, and BC-DSS. The smaller standard deviation value (± 0.74) shows that there is a consistent and stable performance over repeated simulation runs. Moreover, the p-values less than 0.05 prove that the increase in reliability is statistically significant and is not the result of a random variation. Whereas, the existing methods achieves the higher standard deviation of ± 1.88, ± 1.42, and ± 1.16, respectively.

Table 4. Statistical validation of reliability performance

Technique	p-value	Reliability (mean±Std)
EVM-ECC	0.031	82.00 ± 1.84
DeBlock-Sec	0.024	87.00 ± 1.42
BC-DSS	0.015	91.00 ± 1.16
Proposed DRAGO-SLICE	0.004	98.00 ± 0.74

Figure 7 (in Appendix) illustrates the performance of proposed algorithm with existing technique by data integrity, data availability, and energy consumption parameters. Figure 7(a) shows the data integrity performance of EVM-ECC, DeBlock-Sec, BC-DSS, and the proposed method at various data sizes. The proposed DRAGO-SLICE method consistently achieves the highest integrity of 98% as the size of the data increases because of the effective encryption and dual-chunk redundancy measures. BC-DSS demonstrates a moderate improvement when the data size increases, but DeBlock-Sec and EVM-ECC have lower and more moderate values of integrity. Figure 7(b) provides the comparison of the availability of data using four techniques. The proposed DRAGO-SLICE method is able to achieve maximum availability of data up to 94% because of its dual-chunk redundancy and effective slicing plan. BC-DSS and DeBlock-Sec demonstrate moderate improvements with availability ranging between 89% and 92%, and EVM-ECC has the lowest availability in all the rounds. Figure 7(c) illustrates the comparison of energy consumption using four techniques such as EVM-ECC, DeBlock-Sec, BC-DSS, and the proposed method. The proposed DRAGO-SLICE has the lowest energy consumption of 38%, as compared to EVM-ECC, DeBlock-Sec and BC-DSS, which consume a lot of energy. This indicates that the suggested mechanism is more energy-efficient since the encryption and redundancy process is optimized.

5. CONCLUSION

In this paper, a novel DRAGO-SLICE framework has been suggested to improve reliability and security in DCS. The suggested framework achieves the security data protection, and improved reliability through the DJA and DCR. The proposed technique has been analyzed using a Python simulator. As per the comparative analysis, the security of the suggested DRAGO-SLICE model is improved by 21.05%, 13.74%, and 8.52% as compared to the existing EVM-ECC, DeBlock-Sec, and BC-DSS models, respectively. The proposed model improves the security and reliability but still faces the limitation such as high computational costs and low processing device. To counter these limits, the future work will concentrate on real-world cloud testbeds like the AWS or Arizona to determine its scalability and robustness under the realistic network environment, and also integrate key management protocols based on blockchain to provide secure and decentralized distribution of keys in the DRAGO-SLICE framework.

ACKNOWLEDGMENTS

The author would like to express his heartfelt gratitude to the supervisor for his guidance and unwavering support during this research for his guidance and support.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Karuppasamy Lakshmanan	✓	✓		✓			✓			✓		✓		
Vasudevan Venkatraman			✓		✓	✓		✓	✓		✓		✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

INFORMED CONSENT

The undersigned certifies that the nature and purpose of this study have been explained to the above-named individual, including the potential benefits of participation. The questions the individual had about this study have been answered, and we will always be available to address future questions.

ETHICAL APPROVAL

The research supervisor reviewed and ethically approved this manuscript for publication in this journal.

DATA AVAILABILITY

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

REFERENCES

[1] T. V. Doan, Y. Psaras, J. Ott, and V. Bajpai, "Toward Decentralized Cloud Storage With IPFS: Opportunities, Challenges, and Future Considerations," *IEEE Internet Computing*, vol. 26, no. 6, pp. 7–15, Nov. 2022, doi: 10.1109/MIC.2022.3209804.

[2] A. J. Prabhu *et al.*, "Efficient Data Security Using Hybrid RSA-TWOFISH Encryption Technique on Cloud Computing," in *International Conference on Frontiers of Intelligent Computing: Theory and Applications*, vol. 371, 2023, pp. 495–502, doi: 10.1007/978-981-99-6706-3_42.

[3] M. I. Khalid *et al.*, "A Comprehensive Survey on Blockchain-Based Decentralized Storage Networks," *IEEE Access*, vol. 11, pp. 10995–11015, 2023, doi: 10.1109/ACCESS.2023.3240237.

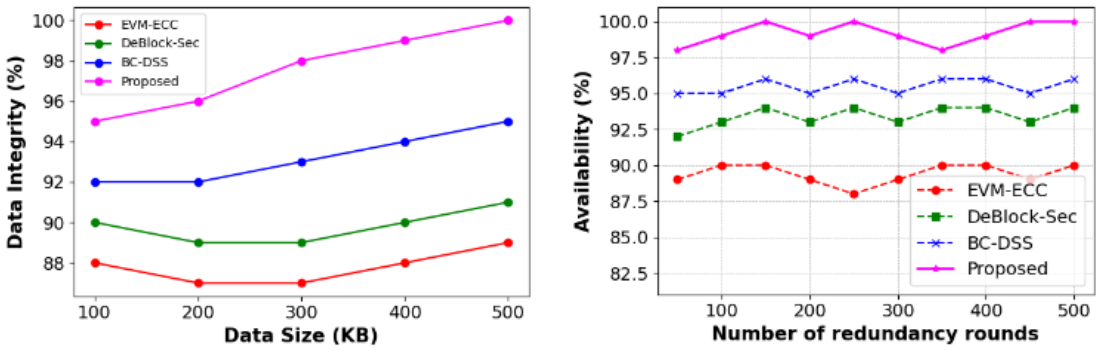
[4] M. Jesi, A. Appathurai, M. Kumaran, and A. Kumar, "Load Balancing in Cloud Computing Via Mayfly Optimization Algorithm," *Revue Roumaine des Sciences Techniques Série Électrotechnique et Énergétique*, vol. 69, no. 1, pp. 79–84, Apr. 2024, doi: 10.59277/RRST-EE.2024.1.14.

[5] M. Rashmi, P. William, N. Yogeesh, and D. K. Girija, "Blockchain-Based Cloud Storage Using Secure and Decentralised Solution," in *International Conference on Data Analytics and Insights*, Singapore: Springer, 2023, vol. 727, pp. 269–279, doi: 10.1007/978-981-99-3878-0_23.

[6] M. Anisha and V. A. Beenu, "Double Secure Cloud Medical Data Using Euclidean Distance-Based Okamoto Uchiyama

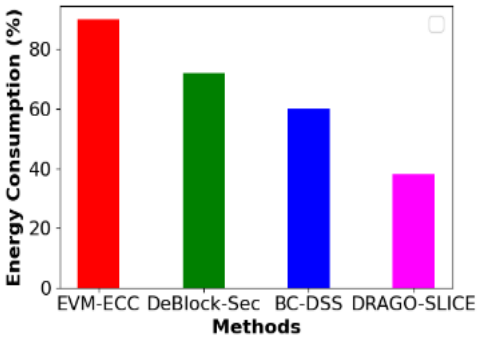
Homomorphic Encryption," *International Journal of System Design and Computing*, vol. 02, no. 01, Jun. 2024, doi: 10.66135/ijsdc0201p001.
[7] S. Ghosh, "Decentralized Storage," in *The Age of Decentralization*, New York: Productivity Press, 2024, pp. 99–117, doi: 10.4324/9781003507352-6.
[8] H. K. Kalidindi, "Crow Search Optimized DNA Encryption for Secure Medical Data Transmission," *International Journal of Computer Engineering and Optimization*, vol. 02, no. 03, pp. 80–85, 2024.
[9] M. Szabó, Á. Recse, R. Szabó, D. Balla, and M. Maliosz, "Separation and optimization of encryption and erasure coding in decentralized storage systems," *Future Generation Computer Systems*, vol. 167, p. 107739, Jun. 2025, doi: 10.1016/j.future.2025.107739.
[10] M. Prabhu, G. Revathy, and R. R. Kumar, "Deep Learning Based Authentication Secure Data Storing in Cloud," *International Journal of Computer Engineering and Optimization*, vol. 1, no. 1, 2023.
[11] A. Aarthy and KV Kanth, "DNA Encryption-Based Secure Access Control and Data Sharing in An Enabled Cloud Environment," *International Journal of System Design and Computing*, vol. 02, no. 01, 2024.
[12] R. Kundu, "On Decentralized Cloud Storage Security and an Efficient Post-Quantum Encryption Scheme," M.S. Thesis, Dept of Electrical and Information Technology, Lund University, Sweden, 2024.
[13] M. M. Merlec and H. P. In, "Blockchain-Based Decentralized Storage Systems for Sustainable Data Self-Sovereignty: A Comparative Study," *Sustainability*, vol. 16, no. 17, p. 7671, Sep. 2024, doi: 10.3390/su16177671.
[14] G. Sreetha and T. V. Chitra, "Block chain assisted cloud based medical data storage via quantum Diffie-Hellman key exchange," *International Journal of Computer Engineering and Optimization*, vol. 01, no. 01, pp. 01–09, 2023.
[15] C. Lee, J. Kim, H. Ko, and B. Yoo, "Addressing IoT storage constraints: A hybrid architecture for decentralized data storage and centralized management," *Internet of Things (Netherlands)*, vol. 25, p. 101014, Apr. 2024, doi: 10.1016/j.iot.2023.101014.
[16] R. Ingle, C. S. K. Selvi, A. Ahilan, N. Muthukumaran, S. Sharma, and M. Kumar, "SERAV Deep-MAD: Deep Learning-Based Security–Reliability–Availability Aware Multiple D2D Environment," *IETE Journal of Research*, vol. 71, no. 2, pp. 523–536, Feb. 2025, doi: 10.1080/03772063.2024.2415502.
[17] H. Zang, H. Kim, and J. Kim, "Blockchain-Based Decentralized Storage Design for Data Confidence Over Cloud-Native Edge Infrastructure," *IEEE Access*, vol. 12, pp. 50083–50099, 2024, doi: 10.1109/ACCESS.2024.3383010.
[18] G. Iovane and R. Amatore, "A Decentralized Storage and Security Engine (DeSSE) Using Information Fusion Based on Stochastic Processes and Quantum Mechanics," *Applied Sciences*, vol. 15, no. 2, p. 759, Jan. 2025, doi: 10.3390/app15020759.
[19] N. Khan, H. Aljoaey, M. Tabassum, A. Farzamnia, T. Sharma, and Y. H. Tung, "Proposed Model for Secured Data Storage in Decentralized Cloud by Blockchain Ethereum," *Electronics*, vol. 11, no. 22, p. 3686, Nov. 2022, doi: 10.3390/electronics11223686.
[20] U. Narayanan, V. Paul, and S. Joseph, "Decentralized blockchain based authentication for secure data sharing in Cloud-IoT: DeBlock-Sec," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 2, pp. 769–787, Feb. 2022, doi: 10.1007/s12652-021-02929-z.
[21] T. Liu, J. Wu, J. Li, J. Li, and Y. Li, "Efficient decentralized access control for secure data sharing in cloud computing," *Concurrency and Computation: Practice and Experience*, vol. 35, no. 17, Aug. 2023, doi: 10.1002/cpe.6383.
[22] R. Mishra, D. Ramesh, S. S. Kanhere, and D. R. Edla, "Enabling Efficient Deduplication and Secure Decentralized Public Auditing for Cloud Storage: A Redactable Blockchain Approach," *ACM Transactions on Management Information Systems*, vol. 14, no. 3, pp. 1–35, Sep. 2023, doi: 10.1145/3578555.
[23] M. Y. Shakor, M. I. Khaleel, M. Safran, S. Alfarhood, and M. Zhu, "Dynamic AES Encryption and Blockchain Key Management: A Novel Solution for Cloud Data Security," *IEEE Access*, vol. 12, pp. 26334–26343, 2024, doi: 10.1109/ACCESS.2024.3351119.
[24] J. Li, Y. Li, J. Wu, Z. Zhang, and Y. Jin, "Blockchain-Based Decentralized Cloud Storage With Reliable Deduplication and Storage Balancing," *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 4, pp. 3289–3304, Jul. 2024, doi: 10.1109/TNSE.2024.3369630.
[25] S. Sharma, S. Sharma, and T. Choudhury, "Enhancing Cloud Data Security Through an Encrypted and Efficient Connection Model based on Blockchain Technology," *Scalable Computing: Practice and Experience.*, vol. 26, no. 2, Feb. 2025, doi: 10.12694/scpe.v26i2.3942.

APPENDIX



(a) (b)

Figure 7. Comparison of: (a) data integrity and (b) data availability



(c)

Figure 7. Comparison of: (c) energy consumption *(continued)*

BIOGRAPHIES OF AUTHORS

Karuppasamy Lakshmanan received his M.E. degree in Computer Science and Engineering from Anna University, Tamilnadu, India, in 2014. He is currently pursuing a Ph.D. degree at Kalasalingam Academy of Research and Education, Tamilnadu, India. His research interests include cloud computing for resource allocation, cloud security, and network. He can be contacted at email: karuppasamy40@outlook.com.

Vasudevan Venkatraman received a Ph.D. degree in 1992. He is currently serving as Registrar and Senior Professor at the School of Computing and Information Sciences, Kalasalingam Academy of Research and Education, Krishnankovil, India. He has published 70 papers in international journals and conferences. He has a large volume of publications in refereed journals. He is a lifetime member of the Indian Society of Technical Education (ISTE). He can be contacted at email: vasudevan82v@outlook.com.