Archives available at journals.mriindia.com

International Journal on Advanced Computer Theory and Engineering

ISSN: 2319 - 2526

Volume 15 Issue 01s, 2026

Optimized Big Data Storage and Security in Cloud Computing using Advanced Encryption Techniques

[1]Mrs. Anuja R. Tungar, [2]Dr. D. S. Deshpane

[1]CSE Department, MGM University, Ch. Sambahjinagar, Maharashtra, India

[2]CSE Department, MGM University, Ch. Sambahjinagar, Maharashtra, India

Email: [1]atungar@mgmu.ac.in , [2]ddeshpande@mgmu.ac.in

Peer Review Information

Submission: 08 Dec 2025

Revision: 25 Dec 2025

Acceptance: 10 Jan 2026

Keywords

Burrows-Wheeler Transform, Binary Banyan Tree Optimization, Big data, Rivest-Shamir-Adleman, Cloud computing

Abstract

Cloud security and effective cloud distribution across the network are the primary issues with cloud computing, but data security is a critical challenge for contemporary society. Advanced encryption techniques are used to secure sensitive data and guarantee privacy in a distributed setting significantly enhancing storage and big data security in cloud computing. Therefore, the input data is collected from the Big Data Derby Dataset. Then, lossless compression is applied using the Burrows-Wheeler Transform (BWT) model to minimize redundancy and enhance cloud-based data accessibility. The lossless compressed data is fed into an Enhanced Rivest-Shamir-Adleman Algorithm (ERSA) to improve data encryption and decryption performance as an optimized Modified Binary Banyan Tree Optimization (BBTO) algorithm is utilized to minimize the time involved for encryption and decryption processes. The proposed Big Data storage and security in cloud computing using advanced encryption techniques (BDSS-CC-AET) technique is implemented on the Python platform, and its performance is expected to outperform existing methods with an estimated high throughput (78mbps), high packet delivery ratio (93%) and low encryption time (381ms). Overall, the proposed BDSS-CC-AET is expected to demonstrate superior effectiveness in big data storage and security in cloud computing.

Introduction

The massive volume, velocity and variety of data gathered every day in the time of digital transformation have made big data storage in cloud environments increasingly significant. Individuals, along with organizations, utilize cloud computing to store and manage both unstructured and structured data. Cloud storage is a popular option for big data management because it provides several benefits, such as scalability, cost-effectiveness, accessibility and flexibility [1]. However, there are key security risks associated with storing private and sensitive data on the cloud, including illegal access, cyber-attacks, data breaches, and privacy violations. To assure data security, confidentiality and integrity, robust encryption and decryption protocols must be developed [2]. Public key and private key cryptography are used by cloud computing systems that provide services to Internet users in order to protect customer data. A common asymmetric key cryptosystem that aims to increase cloud computing security is Elliptic Curve Cryptography (ECC). Compared to non-ECC, ECC offers a number of advantages, such as the ability to use shorter keys with the same strength [3].

Cloud storage is a popular tool for handling massive amounts of both organized and unstructured data in the age of digital transformation. However, serious security

issues such as illegal access, cyber attacks, data breaches and privacy violations take on by this prevalent use. The confidentiality and integrity of sensitive data stored in the cloud are seriously threatened by these problems. Therefore, to improve data protection and provide secure and cloud computing it is essential to adopt strong encryption techniques like ECC.

To prevent unauthorized access and data breaches, cloud storage systems must also have robust security features. Thus, to increase processing speed and security in cloud storage environments, an improved encryption technique and an efficient large data model are needed. The need for improved data organization and security processes is being demonstrated by the big data and cloud computing industries explosive growth. Due to increased accessibility, traditional encryption techniques, such as RSA, have become inefficient and slow, specifically when processing big datasets. This emphasizes the necessity of faster and better encryption methods to secure private data. This research provides a novel approach for effective and secure cloud-based data processing through the use of the BWT for lossless compression, the modified BBTO to reduce encryption/decryption time, and the ERSA algorithm for enhanced data security. The major contributions of this research are summarized as follows:

- Advanced encryption approach is used to improve input integrity in cloud computing that guarantees the improvement of big data derby datasets.
- The BWT model reduces data redundancy, allowing for faster data access and optimum storage use without sacrificing data integrity.
- The ERSA algorithm, utilizing a dual-key modular exponential technique that involves key generation, encryption using two public keys, and decryption using corresponding private keys, is used to improve encryption and decryption and ensure data confidentiality.
- RSA is optimized with a modified BBTO algorithm to speed up the encryption process although preserving strong security. This improves computational efficiency and cuts down on processing time.

The remainder of the research is constructed as follows: a literature survey is presented in Section 2, materials and procedures are discussed in Section 3, results and discussions are described in Section 4, and conclusion is presented in Section 5.

Literature Survey

The process of accessing, managing, and paying for computer resources during the shift from investment to operating costs is improved by cloud computing. Although cloud computing improves cost-effectiveness, control and accessibility, it also presents serious security issues about integrity, confidentiality and authentication [3]. Shivaramakrishna et al. [4] developed a hybrid cryptographic system that utilizes RSA, adaptive management of keys, limited in time control of access, and AES-OTP to increase cloud data security. The systems dynamic management and encryption improve data security, but it also makes the system more complex and challenging to implement. Thabit et al. [5] proposed a two-layer, lightweight homomorphic cryptographic technique to improve cloud computing data security. The method's dual-layer encryption frequently results in increased processing time and resource consumption, but it enhances cloud security by combining symmetric and asymmetric cryptography. Shakor et al. [6] utilized blockchain technology and dynamic AES encryption keys are utilized in a two-phase process to improve cloud data security. Although the usage of blockchain with ECC increases computational complexity and resource requirements, it improves file-level security by utilizing blockchain storage and unique keys. Bouleghlimat et al. [7] addressed the challenges of securing big data in cloud storage using conventional encryption techniques and proposes a novel security strategy to protect data privacy while ensuring effective cloud-assisted big data processing. Compared to existing encryption techniques that are both scalable and effective for big data velocity, the proposed approach balances processing efficiency with improved cloud storage security. Khan et al. [8] discussed the importance of an effective and secure ECC robust security and low overhead existing techniques still used performance security trade off and vulnerabilities.

Cloud computing enhances cost-efficiency and accessibility but raises critical security concerns especially regarding data integrity and confidentiality. Existing solutions such as hybrid cryptography, homomorphic encryption and ECC base blockchain methods improve security but suffer from high complexity, resource consumption. ECC that secure often introduces overhead and vulnerabilities. To overcome these limitations this work proposes the use of an ERSA technique to achieve stronger more efficient and scalable cloud data security without relying on ECC.

Proposed Methodology

The proposed methods are supposed to improve processing speed and big data integrity, making it easier to handle large amounts of data in cloud storage environments. Compared to traditional RSA methods, the use of the ERSA algorithm within the new BBTO framework is expected to provide quicker encryption and decryption, along with better security, reducing the overhead typically associated with cloud storage. Fig. 1 shows the proposed system architecture.

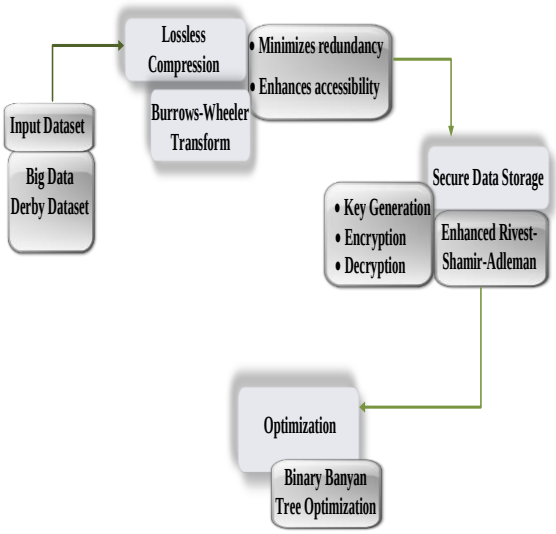

Fig 1: Block diagram of the proposed system

A. Lossless compression using Burrows-Wheeler Transform

The standard BWT's standard technique, represented by the numeral BWT and initially introduced by Burrows and Wheeler [8], is displayed in Algorithm 1. Valid input sequences are able permuted invertible using the BWT transformation of data. The resulting permutation is perfect for further processing using adaptive data compression methods because it is typically globally heterogeneous yet locally homogeneous. A valid input pattern initiates the process. After that, every conceivable rotation of this sequence is created and sorted in lexicographic order. The final column, which contains the last character of each sorted rotation, is extracted from the resulting matrix. The BWT of the original sequence is represented by the final column of symbols.

Algorithm 1: BWT

1: **function** bwt(S)
2: $M \leftarrow$ a matrix including each unique rotation of S
3: $M' \leftarrow$ Arrange the rows of M lexicographically
4: $L \leftarrow$ receive the sorted matrix's final column.
5: **return** L
6: **end function**

B. Secure Data Storage Using Enhanced Rivest-Shamir-Adleman Algorithm

The Enhanced RSA algorithm [9] enables two individuals, a (private key) and b (public key) are needed for secure exchange of data or converse. Algorithm 1 describes in detail the method to generate public and private key. The public file's key is being released through each participant. The public key of the recipient needs to be retrieved from the public folder by the sender in order to compress the file. Provide more details on the encryption procedure, consider Algorithm 2. Once the file has been safely encrypted, the sender sends data for the recipient. The recipient at the receiving end performs the procedures described in Algorithm 3 to decrypt the file utilizing a private key. The steps above describe the ERSA encryption algorithm, including the generation and verification of both public and private keys. The message is encrypted utilizing the receiver's public key after the keys have been confirmed to have been generated correctly. This ensures that a message has to be decrypted and understood through the intended recipient and has the corresponding private key.

Algorithm1. RSA key generation

The procedure for creating encryption and decryption keys in the RSA algorithm can be summarized as follows:

Step 1: Choose two distinct prime numbers, denoted as q *and* p

Step 2: Find the two prime numbers' object $n = qp$

Step 3: Calculate Euler's totient of n and p let $\phi(n) = (q-1)(p-1)$

Step 4: select an integer c to such that $1 < c < \phi(n)$ and gcd $(\phi(n), c) = 1$

Step 5: determine the private exponent e such that $e = g^{-1} \bmod \phi(n)$

Step 6: release the public key as $[g, n]$

Step 7: store the secret key safely as $[e, n]$

Algorithm2. RSA encryption

The data is encrypted and sent to the recipient by the sender

Step 1: A sender receives the recipient's public key
Step 2: A positive integer is created from the message (plaintext). Value m, $1 < m < n$.
Step 3: The message received is encrypted m as $c = m^2 \bmod n$
Step 4: The sender used his private public key to sign the paper, ensuring its reliability
Step 5: The encrypted communication is sent to the recipient by the sender

Algorithm3. RSA decryption
The recipient obtains the encrypted data and uses a secret key to decrypt it restoring it to its original form. The following steps outline the process of decoding the encrypted message.
Step 1: Applying a private key the receiver $[e, n]$ to the cipher text $M = C^g \bmod n$
Step 2: Convert the numeric value from step 1 into its corresponding ASCII character.
Step 3: The authenticity of the message is verified by employing the sender's public key.
Step 4: The final restored message corresponds to the converted value obtained in step 2.

C. Optimization Of Rsa With The Modified Binary Banyan Tree Optimization (Bbto) Algorithm
An innovative meta-heuristic algorithm called the BBTO [10] emerged in inspiration from the traits and development mechanism of banyan trees. The banyan tree is a rare tropical and subtropical plant that is able to grow laterally through several trunks and aerial prop roots to cover a wide region. A banyan tree that is used as an individual indicator in BBTO can divide into several trunks each with difficult leaves and branches that grow around the resource with the support of root systems and heavy branches. To constantly attract high-quality materials and accelerate its growth, the banyan tree makes use of several trunks, branches, and aerial roots.

Step 1: Initialization
In BTGO, an initial branch population Y contains N members. Each member corresponds to set of continuous solutions with M dimensions and the i^{th} members are represented in Eq. (1).

$$Y_i = \left[y_{i1,} y_{i2}, \ldots, y_{ij,}, \ldots, y_{iM} \right] \; 1 \le i \le N, 1 \le j \le M \quad (1)$$

During the initiation stage, the values of the branch population are randomly generated within the search boundaries. The j^{th} dimension of the i^{th} member is computed as expressed in Eq. (2).

$$y_{ij} = y_{\min,j} + rand \times \left(y_{\max,j} - y_{\min,j} \right) \quad (2)$$

Where $rand$ denotes a random value selected from the interval [0, 1], while $y_{\max,}$ and $y_{\min}$ indicate the maximum and minimum limits of the solution space for the j^{th} dimension.

Step 2: Random Generation
The input parameters are initialized and then produced at random. Based on the obvious hyper parameter situation, the optimal values for fitness have been selected.

Step 3: Rooting operator
Banyan aerial roots dig into the ground to extract nutrients in high-humidity regions. The branches that encircle the plants receive nourishment and accelerate their growth with the aid of these aerial roots. These places point to areas that need more resources. Aerial root position is the historical ideal answer that people have looked for in BBTO. The rooting operator is shown in Eq. (3).

$$y_i = P_i^{root} + F \times \left(2 \times rand\left(1, D\right) - 1 \right) \times \left(y_i - P_i^{root} \right) \quad (3)$$

Where P_i^{root} represents the $i-th$ branch explores for this root, $rand\left(1, D\right)$ is a D - dimensional vector of random integers in the interval $[0,1]$ and the frequently constant growth factor is denoted by F.

Step 4: Multi-trunk operator
The banyan tree's trunk also influences branch growth. In addition to providing a stable foundation position for the branches, the thick trunk stores the present ideal resources. Banyan trees possess several trunks, and depending on the nutrient availability in the regions in where they grow, the branches connected to each trunk are able to grow actively. The multi-trunk operator is defined as Eq. (4).

$$y_i = P_{id_i}^{trunk} + F \times \left(2 \times rand\left(1, D\right) - 1 \right) \times \left(P_{id_i}^{trunk} - P_i^{root} \right) \quad (4)$$

Where $P_{id_i}^{trunk}$ represents the *trunk* position associated, $2 \times rand$ represents the transforms the random numbers generated to be in the range of $-1 to 1$, id_i represents the collection of trunks clusters in which the $i-th$ individual is located and P^{trunk} indicates the member of a trunk cluster is best fitting.

Step 5: Adjustment operator
The branches that are close to a water supply and receive enough sunshine have easier access

to nutrients, enabling the branches to grow quickly. Trees naturally gravitate toward light and water for optimal growth, and growth hormones allow trunks in difficult growth conditions to shift toward higher quality resources, as shown in Eq. (5).

$$y_i = r \times y_i + (1-r) \times P_{c_r}^{trunk} \quad (5)$$

Where $P_{c_r}^{trunk}$ represents the cluster selected by m existing subgroups and c_r is the randomly

Chosen group value from $0,1\ldots,m$

Step 6: Termination

Here, the BBTO algorithm optimizes the RSA parameter for accurately big data storage and security in cloud computing with high throughput. Iterating from step 3 to step 6, the procedure continues until the termination condition, $Y = Y + 1$ is reached. Then there's the accurate storage and security of big data in cloud computing utilizing advanced encryption techniques. Fig.2 shows the Flow chart of BBTO Algorithm.

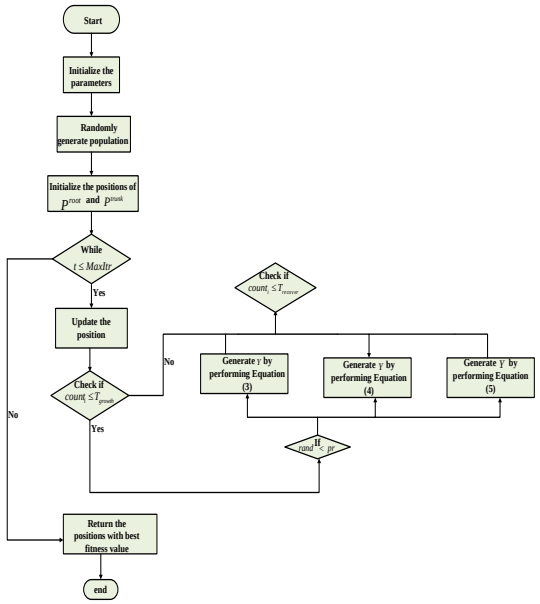

Fig. 2: Flow chart of BBTO Algorithm

Result and Discussion

This section outlines the expected performance of the proposed technique that is implemented in Python for simulation purposes to evaluate its potential impact. This method integrates lossless compression ERSA encryption and the BBTO algorithm to enhance the efficiency and security of large scale data storage in cloud environments. Lossless compression reduces data size while preserving integrity, thereby improving storage and transfer speeds. ERSA employs dual public and private key pairs for enhanced encryption, while the modified BBTO algorithm optimizes encryption performance by reducing processing time. These methods work well together to provide a dependable and effective method to manage massive data sets in cloud-based systems.

A. Dataset Description

The Big Data Derby dataset is public as a result of the New York Racing Association's (NYRA) data science competition. The objective is to motivate participants to produce ideas about horse racing, specifically in relation to strategy, performance, and equine safety, using data and machine learning. The generated dataset is randomly spitted as 80% training, 10% testing, 10% validation.

B. Performance metrics

To evaluate the performance of BDSS-CC-AET method's efficacy and dependability, the metrics such as execution time, encryption time, decryption time, throughput and packet delivery ratio are utilized. Additionally, the BDSS-CC-AET method is compared with other existing methods including IET-SDSC-BFA [3], HCF-SDCC-RSA [4] and ELHCA-DS-CC [5], to assess its effectiveness.

Throughput

The effectiveness of the algorithm is assessed by the throughput rate that is directly proportional to the algorithm's performance the higher performance / the higher throughput. The throughput is calculated using Equation (6).

$$Throughput = PlainText / EncodingTime \quad (6)$$

Encryption Time

The amount of time required to convert plaintext using an encryption algorithm into ciphertext. The encryption time is calculated using Equation (7).

$$Encryption\,Time = \frac{SizeofPlaint\,ext(S)}{Encryption\,Throughput\,(T_{enc})} \quad (7)$$

Where S represents the value of input data and T_{enc} denotes the encryption speed.

Decryption Time

The amount of time required using a decryption algorithm to recover plaintext from ciphertext. The decryption time is calculated using Equation (8).

$$Decryption\ Time = \frac{SizeofCiphertext(C)}{Decryption\ Throughput\ (T_{dec})} \quad (8)$$

Where C represents the encrypted data size and T_{dec} denotes the decryption speed.

Execution time

System execution time is the amount of time it takes finished or executes out a certain task. It is defined in Equation (9).

$$ExecutionTime = I * CPI * T \quad (9)$$

Where I represents the program's number of instructions, CPI is cycles per instruction on average and T denotes the clock cycle time.

Packet Delivery Ratio

A data packet's percentage of successfully transmitted packets from the source to the destination is known as its PDR. The PDR is calculated using Equation (10).

$$PDR = \frac{TotalPackets\,\mathrm{Received}}{TotalPacketsSent} \times 100\% \quad (10)$$

C. Performance analysis

Figures 3-7 show simulation results of the SLA-UEO method. The performance metrics are analyzed by incorporating existing IET-SDSC-BFA, HCF-SDCC-RSA and ELHCA-DS-CC methods.

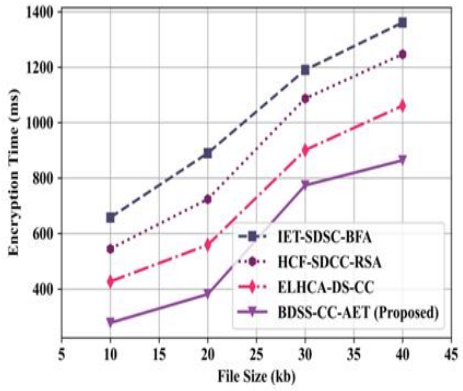

Fig. 3: Performance of Encryption Time Analysis

Fig. 3 shows the performance of encryption time analysis. As the size of the file increases, also rises the encryption time. Using the proposed approach, an expected encryption time of 278 ms is estimated, whereas IET-SDSC-BFA achieved 658 ms, HCF-SDCC-RSA recorded 545 ms, and ELHCA-DS-CC reached 427 ms. Although, encryption time generally increases with the number of files, the proposed BDSS-CC-AET model consistently demonstrates lower encryption time compared to the existing models, namely IET-SDSC-BFA, HCF-SDCC-RSA, and ELHCA-DS-CC.

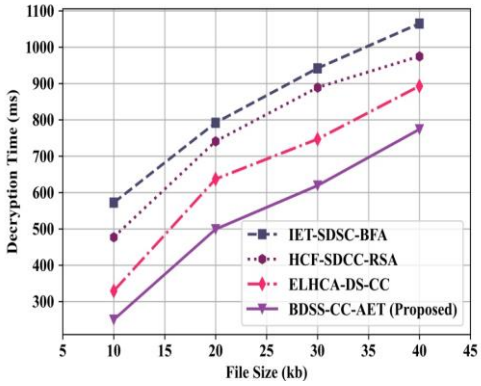

Fig 4: Performance of Decryption Time Analysis

Fig. 4 shows the performance of decryption time analysis. If the files are larger, the decryption time is also higher. For example, IET-SDSC-BFA achieved 572 ms, HCF-SDCC-RSA achieved 477 ms, ELHCA-DS-CC achieved 329 ms, and the proposed approach achieved 250 ms. the present research shows that with increasing file sizes, the proposed BDSS-CC-AET model decrypts data faster than the existing IET-SDSC-RSA, HCF-SDCC-RSA, and ELHCA-DS-CC models. Since encryption keys must be generated prior to file encryption, all models exhibited shorter decryption times compared to encryption times. This is because key generation and initialization processes are not repeated during decryption, reducing the computational load.

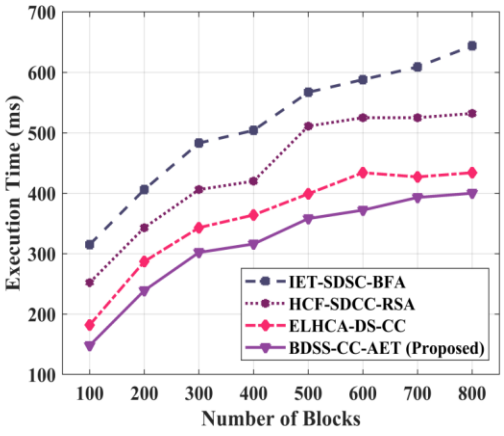

Fig. 5: Performance of Execution Time Analysis

Fig. 5 shows the performance of execution time analysis. The proposed BDSS-CC-AET algorithm continually provides the lowest execution time 148 ms, indicating more efficiency. At the current stage IET-SDSC-BFA exhibits the highest execution time at 315, followed by HCF-SDCC-RSA with 252, and ELHCA-DS-CC with 182. Although it is expected that execution time increases as the number of blocks increases, the

proposed approach shows improved scalability for larger datasets in cloud security applications.

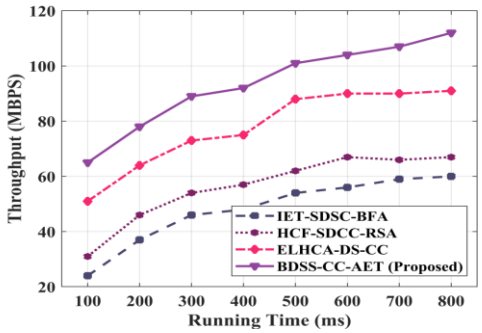

Fig 6: Performance of Throughput Analysis

Fig. 6 shows the performance of throughput analysis. The proposed BDSS-CC-AET consistently achieves the highest throughput across all measured intervals, reaching approximately 65 MBPS, indicating superior performance in big data handling compared to the other techniques. ELHCA-DS-CC shows the second-best performance, followed by HCF-SDCC-RSA and IET-SDSC-BFA. The results indicate that the advanced encryption techniques employed in the proposed BDSS-CC-AET offer significant advantages in terms of data processing and transfer efficiency, thereby enhancing big data storage and security in the cloud.

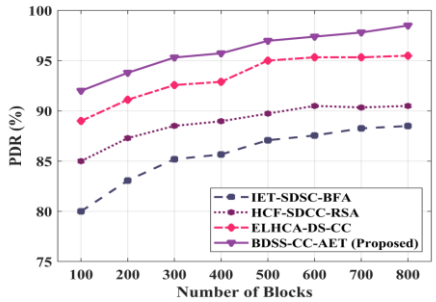

Fig. 7: Performance of Packet Delivery Ratio Analysis

Fig. 7 shows the performance of PDR analysis. The BDSS-CC-AET methodology's data splitting and handling structure result in a greater PDR. Because the IET-SDSC-BFA, HCF-SDCC-RSA, and ELHCA-DS-CC approaches retain the proper structure, the network shows a decrease in PDR. The BDSS-CC-AET method successfully delivers 92% of packets under simulation conditions, compared to 80% for IET-SDSC-BFA, 85% for HCF-SDCC-RSA, and 89% for ELHCA-DS-CC. These results underscore the superior data transmission reliability and network efficiency of the proposed BDSS-CC-AET model in cloud-based big data environments.

Conclusion

This research proposes a novel design that is expected to improve the functionality of encryption and big data security in cloud computing environments. The BDSS-CC-AET technique enhances the BWT and the ERSA algorithm it reduces redundancy and processing time. The optimized BBTO algorithm further accelerates encryption and decryption processes. This approach significantly improves the performance of cloud-based data systems, ensuring both security and accessibility. The user experience in a cloud federation environment uses a throughput of 78mbps, a high PDR of 93% and exhibits a remarkably low encryption time 381ms of additionally, compared to other approaches such as IET-SDSC-BFA, HCF-SDCC-RSA and ELHCA-DS-CC, the proposed BDSS-CC-AET method achieves better convergence. Future research should explore the utilization of homomorphic encryption for secure cloud data processing and the development of decentralized storage solutions to improve data integrity and adherence to data security laws. Furthermore, protecting big data in cloud environments is possible enhanced by implementing AI-powered security techniques.

References

[1] A.E. Adeniyi, K.M. Abiodun, J.B. Awotunde, M. Olagunju, O.S. Ojo, and N.P. Edet, "Implementation of a block cipher algorithm for medical information security on cloud environment: using modified advanced encryption standard approach," *Multimedia Tools and Applications*, Vol. 82, No. 13, pp. 20537–20551, 2023.

[2] V. Krishnasamy and S. Venkatachalam, "An efficient data flow material model based cloud authentication data security and reduce a cloud storage cost using Index-level Boundary Pattern Convergent Encryption algorithm," *Materials Today: Proceedings*, Vol. 81, pp. 931–936, 2023.

[3] B. Seth, S. Dalal, V. Jaglan, D.N. Le, S. Mohan, and G. Srivastava, "Integrating encryption techniques for secure data storage in the cloud," *Transactions on Emerging Telecommunications Technologies*, Vol. 33, No. 4, p. e4108, 2022.

[4] D. Shivaramakrishna and M. Nagaratna, "A novel hybrid cryptographic framework for secure data storage in cloud computing: Integrating AES-OTP and RSA with adaptive key management and Time-

Limited access control," *Alexandria Engineering Journal*, Vol. 84, pp. 275–284, 2023.

[5] F. Thabit, O. Can, S. Alhomdy, G.H. Al-Gaphari, and S. Jagtap, "A novel effective lightweight homomorphic cryptographic algorithm for data security in cloud computing," *International Journal of Intelligent Networks*, Vol. 3, pp. 16–30, 2022.

[6] Sujan Hiregundagal Gopal Rao. (2023). Safety and Security Co-Design in Automotive Semiconductor Systems: Challenges and Future Directions. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(4s), 830–834. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/7999.

[7] 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.

[8] N. Khan, Z. Jianbiao, H. Lim, J. Ali, I. Ullah, M. Salman Pathan, and S.A. Chaudhry, "An ECC-based mutual data access control protocol for next-generation public cloud," *Journal of Cloud Computing*, Vol. 12, No. 1, p. 101, 2023.

[9] L. Cheung, A. Moffat, and C. Rizkallah, "Formalized Burrows-Wheeler Transform," in *Proceedings of the 14th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pp. 187–197, Jan. 2025.

[10] U. Musa, M.O. Adebiyi, F.B. Osang, A.A. Adebiyi, and A.A. Adebiyi, "An improved secured cloud data using dynamic Rivest-Shamir-Adleman key," *Indonesian Journal of Electrical Engineering and Computer Science*, Vol. 33, No. 1, pp. 433–441, 2024.

[11] X. Wu, W. Zhou, M. Fei, Y. Du, and H. Zhou, "Banyan tree growth optimization and application," *Cluster Computing*, Vol. 27, No. 1, pp. 411–441, 2024.

[12] https://www.kaggle.com/competitions/big-data-derby-2022