International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 10 Issue 1, Jan-Feb 2026 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Scalable Encryption Algorithms for Secure Big Data Storage and Transmission

Dr. Gopal Prasad Sharma[1], Prof. Dr. Manish Pokharel[2], Prof. Raj Kumar Thakur[3], Prof. Dr. Pawan Kumar Jha[4]

[1]Associate Professor, Purbanchal University School of Science & Technology (PUSAT), Biratnagar, Nepal
[2]Professor, Department of Computer Science and Engineering, Kathmandu University, Nepal
[3,4]Professor, Purbanchal University School of Science & Technology (PUSAT), Biratnagar, Nepal

ABSTRACT

The rapid growth of big data across industries has led to significant concerns regarding the security of sensitive information. As organizations store, transmit, and process massive volumes of data, ensuring the privacy, integrity, and confidentiality of this data becomes crucial. This paper explores scalable encryption algorithms designed to address the unique security challenges posed by big data environments. We examine various encryption techniques, including symmetric encryption algorithms such as AES, asymmetric encryption methods like RSA and ECC, as well as advanced encryption models such as homomorphic encryption and attribute-based encryption. Additionally, the paper delves into the scalability challenges associated with encrypting large datasets, focusing on computational efficiency, key management, and performance issues in big data contexts. Real-time encryption for secure transmission, as well as emerging trends like quantum-resistant encryption and AI-enhanced encryption processes, are also discussed. Through case studies, we highlight how scalable encryption is applied in sectors such as healthcare, finance, and IoT, underscoring the critical role of robust encryption in securing big data. The paper concludes with an analysis of current challenges and future directions for big data encryption, aiming to offer solutions to meet the growing demands of data security.

KEYWORDS: *Asymmetric Encryption, Attribute-Based Encryption, Big Data, Scalable Encryption, Symmetric Encryption.*

How to cite this paper: Dr. Gopal Prasad Sharma | Prof. Dr. Manish Pokharel | Prof. Raj Kumar Thakur | Prof. Dr. Pawan Kumar Jha "Scalable Encryption Algorithms for Secure Big Data Storage and Transmission" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-10 | Issue-1, December 2026, pp.10-18, URL: www.ijtsrd.com/papers/ijtsrd97164.pdf

IJTSRD98732

I. INTRODUCTION

Large and complex data sets are difficult to manage and analyse using traditional data management technologies. The "three Vs"-volume, velocity, and variety-define it, which often contains organised and unstructured data. The growing volume of data produced by gadgets, social media, sensors, and applications makes big data lucrative for organisations, governments, and corporations. Its useful information drives decisions, operational efficiency, and creativity [1]. Big data helps finance uncover fraud and optimise investment strategies, while healthcare uses predictive analytics to improve patient outcomes. Big data has great potential but poses major privacy and security dangers. Unauthorised access can lead to data breaches and sensitive information disclosure, which is concerning. Data storage in on-premise data centres and cloud platforms makes security, integrity, and availability harder to ensure. Hacking, ransomware, and insider threats allow cybercriminals to access huge data settings. Big data is too large and diverse for standard security measures, thus it needs more secure storage, access control, and data transmission. Encryption is vital for huge data protection. Unauthorised parties cannot view the data without the decryption key since it makes it illegible. Large data sets benefit from encryption to secure private data, financial records, and intellectual property. Encryption prevents data breaches, unauthorised access, and tampering at rest and in transit, ensuring privacy and GDPR and HIPAA compliance [2]. This article discusses scalable encryption to protect enormous data storage and transit. Encryption solutions that can handle big data environments' scale and complexity without affecting performance are essential as data volumes expand. The paper will discuss encryption methods, their scalability issues, and their use in big data security across industries. Understanding these algorithms helps organisations protect their data and provide seamless access and transfer across platforms and locations.

II. BACKGROUND AND FUNDAMENTALS

A. BIG DATA CHARACTERISTICS

The "5 Vs"-volume, variety, velocity, veracity, and value-distinguish big data. Understanding these qualities is necessary before discussing large data encryption. The mean "volume" as the daily amount of data created. The rise of online platforms, social media, IoT devices, and sensor networks has increased data generation. Single organisations can generate terabytes or petabytes of data, making standard data processing technologies unsuitable. Big data requires fast storage and efficient encryption. Diversity is the variety of facts collected from different sources [3]. This category includes relational databases, XML and JSON files, emails, social media posts, videos, and photos. Data processing and encryption are complicated by the fact that different data types require different encryption or storage methods. To demonstrate, audiovisual data may require different encryption than text-based records. Velocity measures data creation, processing, and analysis speed. Data streams in real-time applications like financial transactions and health monitoring demand lightning-fast encryption and decryption. Encryption must be secure and efficient to keep big data systems working properly in real time. Dependability determines information credibility [4]. Large data sets may contain inaccurate, incomplete, or noisy data. It's crucial to verify all data since faulty or unverified data might lead to flawed research and judgements. Encrypting sensitive data prevents unauthorised access and alterations. The practical insights big data provides make it useful. Big data's potential is unlocked by analysing and identifying major trends and patterns. For later usage, this data must be securely stored and transmitted. Data encryption protects sensitive data while allowing authorised access and analysis.

B. ENCRYPTION BASICS

Encryption is vital for data privacy. Symmetric encryption is useful for large datasets since it only needs one key. Asymmetric encryption, which uses a public key for encryption and a private key for decryption, is more secure but slower and more computationally expensive.

Symmetric and asymmetric encryption require key management, especially in large data sets [5]. Key management systems prevent disclosure, distribution, and unauthorised access to secrets. Scalable big data encryption methods are needed to process enormous volumes of data without sacrificing performance. This is critical in distributed systems and databases where data is stored in multiple locations. Traditional encryption can slow down real-time data-intensive applications like financial trading and healthcare. Key management systems must scale to handle enormous amounts of keys when handling massive volumes of data. Big data ecosystems can balance security and operational efficiency with the right encryption and key management solutions.

III. ENCRYPTION ALGORITHMS FOR BIG DATA

A. SYMMETRIC ENCRYPTION ALGORITHMS

Symmetric encryption is popular for protecting large data due to its effectiveness and cheap computing cost. These methods reuse the key for decryption, making them better for encrypting large datasets quickly. However, managing and distributing the encryption key securely may be difficult, especially in large-scale situations. AES, or Advanced Encryption Standard, is a popular symmetric encryption technique. AES is the standard for large-scale data encryption due to its dependability, scalability, and efficiency [6]. The most secure version, AES-256, supports 128-bit, 192-bit, and 256-bit keys. AES's quick processing of enormous data and superior security appeal to the government, healthcare, and financial sectors. One of the first symmetric encryption algorithms, IBM's Data Encryption Standard (DES), was used by the US government in the 1970s. DES was formerly the gold standard for encryption, but brute-force attacks and other computational power threats have diminished its 56-bit key length. RC4, another popular 1990s symmetric encryption method, was utilised in SSL/TLS for communication security. RC4, a stream cypher, encrypts data bit by bit, making it excellent for streaming. The keystream biases in RC4 led to its deprecation for most safe applications. Despite its historical importance, RC4's security and performance make it unsuitable for large datasets. Symmetric encryption's effectiveness and ability to manage enormous data sets with minimum CPU load are crucial in big data environments. Data-at-rest encryption is perfect for quickly and efficiently encrypting massive volumes of data. However, key distribution and administration are more complicated because all parties need the same key [7]. Key management and preventing unauthorised access are growing harder as large data grows.

B. ASYMMETRIC ENCRYPTION ALGORITHMS

Asymmetric encryption (public-key encryption) requires two keys, one public and one private, to encrypt and decrypt data. Asymmetric encryption is more computationally expensive and slower than symmetric encryption, but it can give superior security in some cases. Asymmetric algorithms are used for encryption and digital signatures. Popular asymmetric encryption techniques include RSA (Rivest-Shamir-Adleman).

It uses the difficulty of factoring large prime integers. RSA is safe but computationally intensive, making it unsuitable for direct encryption of large data.

However, symmetric encryption systems often use RSA to encrypt the symmetric key and huge volumes of data [8]. The algebraic nature of elliptic curves over finite fields underpins asymmetric encryption using ECC. ECC is a fast, secure alternative to RSA that employs smaller key sizes, making it suited for low-resource environments. Increasingly, ECC is the preferred method for safeguarding communications on mobile devices, Internet of Things apps, and more. Diffie-Hellman is a cryptographic protocol, not an encryption method, however it is used to protect cryptographic key exchange over unsecured channels. Two parties can agree on a secret key for symmetric encryption. Despite not encrypting data, Diffie-Hellman is vital in large data environments where encrypted communications require safe key exchange. Asymmetric encryption protects tiny data and key exchange well. However, processing large data sets takes longer and requires more computational capacity. Asymmetric encryption is not recommended for large datasets due to performance penalty. The secure exchange of symmetric encryption keys and communication channel protection are its strengths [9].

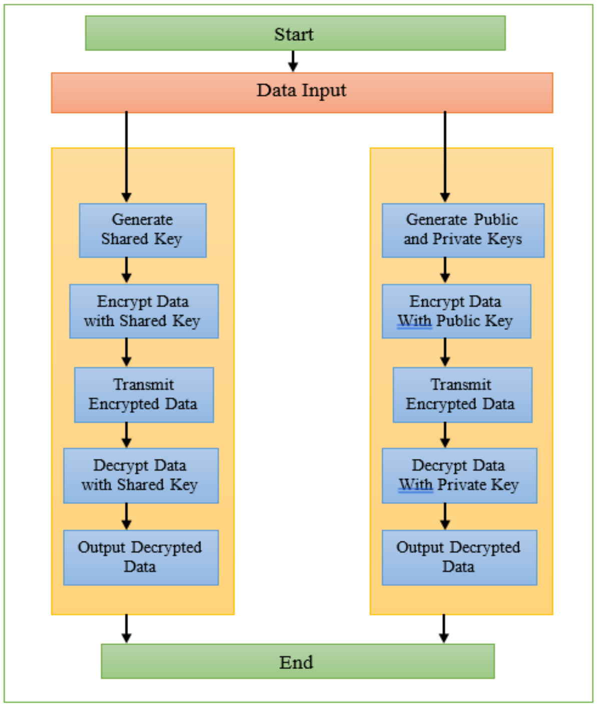
Start
Data Input
Generate Shared Key
Encrypt Data with Shared Key
Transmit Encrypted Data
Decrypt Data with Shared Key
Output Decrypted Data
Generate Public and Private Keys
Encrypt Data With Public Key
Transmit Encrypted Data
Decrypt Data With Private Key
Output Decrypted Data
End

Figure 1 Symmetric vs. Asymmetric Encryption Process Flow diagram

C. HOMOMORPHIC ENCRYPTION

Homomorphic encryption, which does not require decryption, is remarkable. This feature helps secure data privacy while facilitating computations and analytics in massive data sets. A cloud provider can process encrypted data for a client without seeing the original [10]. By encrypting data even while processing, homomorphic encryption for big data greatly reduces exposure risk. Homomorphic encryption is far from being ready for huge datasets. Homomorphic encryption's high processing cost and slow encryption/decryption make it unsuitable for large datasets.

Also, homomorphic encryption is more resource-intensive and complicated to implement than ordinary encryption.

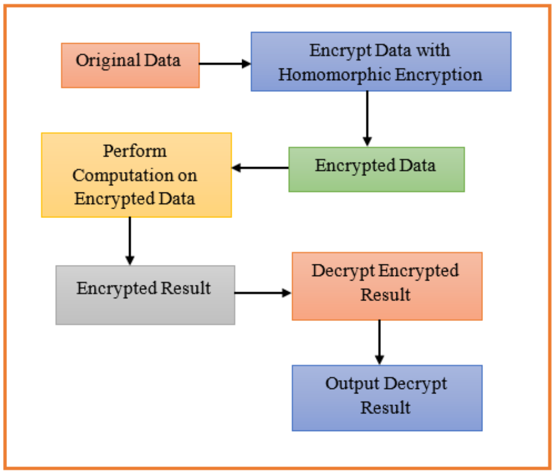

Figure 2 Homomorphic Encryption Block Diagram

D. ATTRIBUTE-BASED ENCRYPTION (ABE)

Modern encryption methods like Attribute-Based Encryption enable precise control over encrypted data. ABE encryption is more versatile and granular since it uses user or data attributes. The data owner can control ciphertext access with CP-ABE [11]. Decrypted user attributes must meet data owner policy after CP-ABE encryption. Alternative approach Key-Policy Attribute-Based Encryption (KP-ABE) links decryption keys using access policies. KP-ABE generates keys based on qualities, so only users with the right attributes may decode data. ABE is ideal for big data because it offers fine-grained access control, which is essential for diverse and sensitive information. ABE can verify that only approved staff can access certain medical documents in healthcare. The processing complexity of encryption and decryption and the difficulties of ABE may cause scalability issues with large datasets.

E. POST-QUANTUM CRYPTOGRAPHY

RSA and ECC, long-standing cryptographic protocols, may be overwhelmed by quantum computing. Quantum computers employ quantum physics to process data differently than classical computers [12]. Existing encryption methods require computing discrete logarithms or factoring large integers, which are difficult. Post-quantum cryptography (PQC) was created to create quantum-resistant encryption methods. Lattice-based, hash-based, and code-based cryptography use mathematical problems that quantum computers may struggle with. Post-quantum cryptography is still young, but it will play a crucial role in securing massive data in the future, especially in long-term storage or data preservation contexts. Implementing quantum-resistant encryption will become increasingly important as quantum computing improves. As computational power increases, we need post-quantum cryptography to secure large data. Before PQC can be extensively used, scalability is unknown, and existing systems and protocols may need major renovations.

IV. SCALABILITY CHALLENGES IN ENCRYPTION

A. SCALABILITY REQUIREMENTS

Scalability in big data encryption means managing expanding data volumes without compromising security or performance. Scalability depends on computer efficiency, speed, storage, and growth adjustability. Real-time or large-scale encryption must be fast and effective to prevent processing overhead and latency as data quantities increase [13]. AES is more efficient and uses fewer resources than asymmetric algorithms. Scalable encryption is crucial in storage-constrained environments because it must limit data size growth and respond to rapidly growing datasets without hardware upgrades or reconfigurations. In dynamic large data environments, horizontal scalability-the capacity to add resources-is essential for performance stability.

B. PERFORMANCE ISSUES

Encrypting huge datasets causes performance concerns that can reduce system efficiency. These issues can hinder scaling encryption for large data. Working with large datasets is slowed by encryption and decryption. The time needed to encrypt or decrypt huge datasets using AES or RSA grows with data capacity. Symmetric encryption is faster than asymmetric, although big data situations can still cause delays owing to the amount of data. In low-latency data access environments, encrypting petabytes of data for storage or transmission can take too long [14]. Big data systems are sensitive to encryption and decryption delay. In real-time applications like financial trading platforms, cloud services, and video streaming, encryption/decryption delays can hurt performance. Encrypting data in a distributed big data context may generate latency and synchronisation issues because it must be dispersed across several nodes or computers. Many big data systems use parallel computing frameworks like Hadoop and Spark to handle enormous datasets. Parallel processing may improve performance, but synchronising encryption and decryption over multiple nodes is difficult. If various nodes have varying encryption and decryption timings, performance constraints can make scaling the encryption process throughout the dataset difficult.

C. KEY MANAGEMENT

Key management becomes exponentially more difficult as the scale of the data grows. In a big data environment, the number of encryption keys required for securing vast amounts of data can be immense.

- Distributing encryption keys safely in large-scale systems, especially in clouds, is problematic due to interruption or compromise. Nodes and users struggle to share keys while data is spread across many locations.
- Big data systems struggle to manage and keep many encryption keys safely. Because of their scaling requirements, Hardware Security Modules (HSMs) and Key Management Systems (KMS) increase complexity when centralising key storage [15].
- Key rotation and expiration are needed to secure large systems. These activities must be automated using key management systems to maintain the system secure and error-free as it grows.

D. LOAD BALANCING AND PARALLELIZATION

Parallelisation and load balancing are necessary for huge data encryption. Multiple ways exist for encryption process parallelisation. Large-scale systems can partition data for independent encryption, speeding up the process. Synchronisation is necessary to preserve dataset data [16]. Parallel encryption: Parallelisation speeds up symmetric techniques like AES when encrypting large datasets, however RSA is harder to parallelise. Optimise parallel encryption for various ways to improve scalability. By extending encryption across multiple nodes, distributed encryption removes bottlenecks and ensures redundancy. Unbalanced load is prevented by proper encryption task management.

V. SECURE STORAGE OF BIG DATA

A. ENCRYPTED DATA STORAGE

To protect sensitive data in big data systems from breaches, unlawful access, and other dangers, encryption is necessary. Security measures like encryption restrict access to previously accessible content to authorised users with the right key. Large amounts of data are encrypted in databases, the cloud, and distributed file systems. Databases often encrypt sensitive data in columns or tables. Oracle Database and Microsoft SQL Server often encrypt storage-level databases via Transparent Data Encryption (TDE). Performance and scalability management are important since encrypting vast quantities of data increases computing overhead and query latency [17]. Amazon S3, Microsoft Azure, and Google Cloud Storage offer scalable solutions but security issues. Encryption before data transfer by the cloud provider or consumers can solve these issues. Client-side encryption is better for high-security requirements since it allows you more control, even though server-side encryption is easier to maintain. Cloud-based KMSs and HSMs help safeguard encryption keys, a vital component. Distributed file systems like HDFS and Apache Cassandra encrypt files or blocks. HDFS data blocks are encrypted to protect critical data if a node is compromised. Encryption and decryption can slow down these systems, which is problematic for data streaming and real-time analytics.

B. DATA REDUNDANCY AND ENCRYPTION

Big data systems use data replication and erasure coding for data availability, fault tolerance, and disaster recovery. This ensures data access after system crashes or hardware failures. These techniques should be used with encryption to secure data. Data replication creates copies of data on many nodes to provide data recovery in the case of a node failure. The more encrypted data in the system can cause scalability challenges, but it improves availability. Encrypting each copy increases computational and storage needs. Key management is more complicated because each copy needs its own encryption key to protect it.

Erasure coding can also divide data into smaller parts, add redundant data, and distribute them across storage nodes [18]. Erasure coding reduces storage expense by reconstructing some data after data loss, unlike replication, which copies the entire dataset. Erasure coding complicates encryption by requiring each piece to be retained and reassembled.

Erasure coding enhances storage efficiency in encrypted big data systems without affecting security or dependability.

C. ENCRYPTION IN DISTRIBUTED SYSTEMS

Big data platforms like Hadoop, Apache Cassandra, and MongoDB use distributed systems to scale horizontally and improve storage capacity and fault tolerance. However, encryption challenges these systems. HDFS-Transparent Data Encryption (TDE) protects Hadoop's HDFS data before writing it to disc. Although this ensures security, it can slow things down because every node must encrypt and decrypt, which is problematic with large databases. Hadoop supports Kerberos authentication and KMIP for cluster-wide encryption key management, however performance issues remain. Apache Cassandra, a distributed NoSQL database, encrypts disc and node data at-rest and in-transit. Similar to HDFS, the computational cost of encrypting and decrypting vast volumes of data over many nodes may slow encryption performance. Key management must be efficient to secure data without losing scalability [19]. Key distribution across the cluster is usually done centrally. Distributed systems' biggest challenge is balancing encryption with efficiency. Data security requires encryption, but it can slow down real-time analytics systems. A workaround is to use GPUs or FPGAs for encryption. This improves system performance without compromising security.

VI. SECURE TRANSMISSION OF BIG DATA

In distributed systems, where data moves between nodes, data centres, and cloud platforms, safe big data transmission is essential to protect sensitive data during network transit. Data of this volume and relevance requires secure techniques and robust encryption. This section discusses data encryption during transmission, real-time encryption, and the challenges of secure transmission without performance degradation.

A. ENCRYPTION IN DATA TRANSMISSION

Big data transmissions must be encrypted to prevent eavesdropping and data manipulation. Asymmetric encryption technologies like TLS and SSL ensure system connectivity, while TLS is recommended for its superior security. VPNs create private, encrypted tunnels for remote data transport [20]. Due to complex data pipelines, high-speed transmission needs, and large data quantities, big data data security requires balancing encryption and system efficiency.

B. REAL-TIME ENCRYPTION

Real-time encryption protects sensitive data travelling between distributed computers. Large data applications require this for real-time analytics and machine learning. Even though real-time encryption secures data, encryption and decryption may slow speed. This problem is often solved with GPUs and FPGAs. Efficient cryptographic algorithms like AES are preferred for high-throughput systems to balance security and efficiency.

C. CHALLENGES IN TRANSMISSION SECURITY

- Big data applications often involve huge volumes of data, leading to increased packet sizes due to encryption overhead.
- This can slow data transfer speeds, so choose encryption options that don't require too much bandwidth but protect sensitive data, like lightweight encryption or compression [21].
- Encryption can slow real-time data processing.
- To maximise performance and data transfer, encryption systems must be optimised to reduce delay.
- TLS/SSL provide excellent security, however large datasets may slow them down. VPNs improve security but slow data flow.
- Optimising encryption methods with hardware offloading and parallel encryption helps balance performance and security.
- The number of infrastructural levels affects data encryption.
- Distributed systems face the challenge of ensuring encryption works across without delaying the system.

VII. CASE STUDIES

A. HEALTHCARE – SECURING PATIENT DATA WITH ENCRYPTION

Healthcare data including prescription lists and medical records must be encrypted. Hospital patient records are encrypted with AES in transit and at rest on the network [22]. Transferring sensitive data like patient records uses TLS/SSL. This may help us avoid data breaches and comply with HIPAA. Scalable encryption is required to secure and speed up big data sets in several places.

B. E-COMMERCE – ENCRYPTING CUSTOMER DATA FOR SECURE TRANSACTIONS

Online retailers use RSA encryption to protect credit card credentials. Another use of homomorphic encryption is analysing client data without decrypting. During busy shopping seasons, managing huge numbers of transactions requires swift scaling up or down. Cloud architecture and parallelisation make transmission encryption efficient and safe [23]. Both use cases demonstrate the importance of scalable encryption in protecting sensitive healthcare and e-commerce data.

VIII. CHALLENGES AND FUTURE DIRECTIONS

The quantity, velocity, and diversity of big data constitute the biggest challenges to scalable encryption. Encryption's processing burden can cripple system performance with large files. Large numbers of encryption keys in distributed systems can bring inefficiencies and security issues, complicating key management. Big data security is complicated enough without having to update and assess encryption protocols for GDPR and HIPAA. Current encryption technologies may struggle to handle data volumes without compromising security. Quantum-resistant cryptography, homomorphic encryption, and artificial intelligence-optimized encryption may solve these problems. Quantum-resistant algorithms are needed to prepare for future quantum computing hazards. However, homomorphic encryption would allow computations on encrypted data, improving efficiency and security. AI and ML could automate key management, improve anomaly detection, and simplify encryption to adapt to large data environments. New advances in scalability, performance, and compliance management could help satisfy large data security demands.

IX. CONCLUSION

The article concluded that scalable encryption is essential for huge data protection. We discussed how huge data's volume, velocity, and variety make encryption challenging, among other qualities. Symmetric (AES, DES), asymmetric (RSA, ECC), homomorphic, and attribute-based encryption were examined for big data security, each with pros and cons. Scalability is crucial for big data algorithms because they must manage enormous amounts of sensitive data efficiently without sacrificing efficiency. We examined how encryption technologies like TLS/SSL can address latency and capacity issues that hinder big data transmission. Blockchain integration, AI-driven encryption advances, and quantum-resistant algorithms could help safeguard massive data in the future. Massive data sets require strong encryption. As data volumes grow, organisations must use scalable encryption to protect data privacy, integrity, and security. Encrypting data helps companies comply with data protection laws and build confidence. To handle the expanding complexity and volume of big data, encryption methods and strategies will evolve. AI and quantum-resistant encryption and continual technology improvement will be needed to secure massive data from new threats.

REFERENCE

[1] S. Atiewi, A. Al-Rahayfeh, M. Almiani, S. Yussof, O. Alfandi, A. Abugabah, and Y. Jararweh, "Scalable and secure big data IoT system based on multifactor authentication and lightweight cryptography," *IEEE Access*, vol. 8, pp. 113498-113511, 2020.

[2] M. N. Ramachandra, M. Srinivasa Rao, W. C. Lai, B. D. Parameshachari, J. Ananda Babu, and K. L. Hemalatha, "An efficient and secure big data storage in cloud environment by using triple data encryption standard," *Big Data and Cognitive Computing*, vol. 6, no. 4, p. 101, 2022.

[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] G. Viswanath and P. V. Krishna, "Hybrid encryption framework for securing big data storage in multi-cloud environment," *Evolutionary Intelligence*, vol. 14, no. 2, pp. 691-698, 2021.

[5] M. A. P. Chamikara, P. Bertók, D. Liu, S. Camtepe, and I. Khalil, "An efficient and scalable privacy preserving algorithm for big data and data streams," *Computers & Security*, vol. 87, p. 101570, 2019.

[6] U. Narayanan, V. Paul, and S. Joseph, "A novel system architecture for secure authentication and data sharing in cloud enabled Big Data Environment," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 3121-3135, 2022.

[7] G. Kapil, A. Agrawal, A. Attaallah, A. Algarni, R. Kumar, and R. A. Khan, "Attribute based honey encryption algorithm for securing big data: Hadoop distributed file system perspective," *PeerJ Computer Science*, vol. 6, p. e259, 2020.

[8] A. Alabdulatif, I. Khalil, and X. Yi, "Towards secure big data analytic for cloud-enabled applications with fully homomorphic encryption," *Journal of Parallel and Distributed Computing*, vol. 137, pp. 192-204, 2020.

[9] Y. M. Essa, E. E. D. Hemdan, A. El-Mahalawy, G. Attiya, and A. El-Sayed, "IFHDS: intelligent framework for securing healthcare bigdata," *Journal of Medical Systems*, vol. 43, no. 5, p. 124, 2019.

[10] A. A. Mir, "Optimizing mobile cloud computing architectures for real-time big data analytics in healthcare applications: Enhancing patient outcomes through scalable and efficient processing models," *Integrated Journal of Science and Technology*, vol. 1, no. 7, 2024.

[11] J. Jayabalan and N. Jeyanthi, "Scalable blockchain model using off-chain IPFS storage for healthcare data security and privacy," *Journal of Parallel and Distributed Computing*, vol. 164, pp. 152-167, 2022.

[12] A. Bahmani et al., "A scalable, secure, and interoperable platform for deep data-driven health management," *Nature Communications*, vol. 12, no. 1, p. 5757, 2021.

[13] I. El Alaoui and Y. Gahi, "Network security strategies in big data context," *Procedia Computer Science*, vol. 175, pp. 730-736, 2020.

[14] D. B. Rawat, R. Doku, and M. Garuba, "Cybersecurity in big data era: From securing big data to data-driven security," *IEEE Transactions on Services Computing*, vol. 14, no. 6, pp. 2055-2072, 2019.

[15] L. Guo, H. Xie, and Y. Li, "Data encryption based blockchain and privacy preserving mechanisms towards big data," *Journal of Visual Communication and Image Representation*, vol. 70, p. 102741, 2020.

[16] M. A. M. Abu-Faraj and Z. A. Alqadi, "Improving the efficiency and scalability of standard methods for data cryptography," *International Journal of Computer Science & Network Security*, vol. 21, no. 12spc, pp. 451-458, 2021.

[17] S. Han, K. Han, and S. Zhang, "A data sharing protocol to minimize security and privacy risks of cloud storage in big data era," *IEEE Access*, vol. 7, pp. 60290-60298, 2019.

[18] J. Chen, L. Ramanathan, and M. Alazab, "Holistic big data integrated artificial intelligent modeling to improve privacy and security in data management of smart cities," *Microprocessors and Microsystems*, vol. 81, p. 103722, 2021.

[19] S. H. Mousavi, M. Khansari, and R. Rahmani, "A fully scalable big data framework for Botnet detection based on network traffic analysis," *Information Sciences*, vol. 512, pp. 629-640, 2020.

[20] Z. Ma, L. Wang, X. Wang, Z. Wang, and W. Zhao, "Blockchain-enabled decentralized trust management and secure usage control of IoT big data," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4000-4015, 2019.

[21] A. Tchernykh et al., "Scalable data storage design for nonstationary IoT environment with adaptive security and reliability," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10171-10188, 2020.

[22] M. Akbar, I. Ahmad, M. Mirza, M. Ali, and P. Barmavatu, "Enhanced authentication for de-duplication of big data on cloud storage system using machine learning approach," *Cluster Computing*, vol. 27, no. 3, pp. 3683-3702, 2024.

[23] S. P. Gochhayat, E. Bandara, S. Shetty, and P. Foytik, "Yugala: Blockchain based encrypted cloud storage for IoT data," in *2019 IEEE International Conference on Blockchain (Blockchain)*, 2019, pp. 483-489.