Performance Comparison of Cryptographic Algorithms for Big Data Stored in Cloud

Jung Kyu Park[1], Eun Young Park[2*]

[1]Department of Computer Engineering, Changshin University, Changwon-si, Korea, jkpark@cs.ac.kr
[2]Rinascita College of Liberal Arts and Sciences, Shinhan University, Uijeongbu-si, Korea, eypark@shinhan.ac.kr

Due to the rapidly evolving information and communication technology, newly created data is rapidly increasing. Data from sensors, smartphones, social networking sites, logical data, and initiatives are fueling this massive information explosion. Hadoop is a framework used to process vast amounts of data. Recently, Hadoop has been widely used in businesses to store and manage important information. Hadoop is a distributed file system that is constantly being updated. However, Hadoop is vulnerable to the security of sensitive data. In this study, security issues for cloud systems were investigated. In particular, encryption algorithms available in the current cloud environment were investigated and data encryption experiments were conducted. Through comparative experiments of various algorithms, an algorithm was selected and used in a cloud environment. The algorithm used in this study can be used for data protection in the Hadoop environment. Based on experiments, it has been shown that the suggested framework with Salsa20 has the biggest avalanche effect, the lowest run time, and the maximum throughput.

Keywords: Cloud, Encryption, HDFS, Hadoop, SLASA20.

1. Introduction

Every day, a considerable amount of data is generated. Data from transactions, social media companies, mobile devices, sensors, and public sources are added to this deluge of data. The reason for this unanticipated inflow is that we have created more information than ever in the last five years. Big Data, which refers to these pervasive huge bits of information, has emerged as one of today's most hotly debated research trends. Enormous Data is the collection of a large amount of information, which can take any shape, including ordered frames and unstructured frames. The production of big data has also resulted in the application of machine learning algorithms for better and rapid analysis. Its advantages can be many. The performance comparison of cryptographic algorithms for big data in the cloud requires careful consideration of security and efficiency aspects which can be enhanced by machine learning in threat detection and optimization. For example, ML integration with

animal and plant research advances biodiversity conservation and ecological understanding (Alzuni, 2023; Cho et al., 2024). In the business domain, AI drives data-driven decision-making, personalized services, and operational efficiencies, showcasing its transformative potential across diverse sectors (Vainshnav & Dave, 2022).

Due to its storage capacity for organized and unstructured, social and non-social material, it is highly known in a few disciplines. It is an opportunity to improve business for large associations and corporate developments. Figure 1 illustrates the prediction for big data revenue in America (in billions of dollars), which highlights the significance of big data going forward (Taylor, 2022; Sandhu, 2022).

Data is provided in wide aggregates due to correspondence and data transfer, and the Big Data need to have been prepared for information mining calculations. Effective frameworks must be created in order to fully utilize this opportunity while taking into account the present difficulties given by its size, structure, and security. Today's engineering of information preparation frameworks has seen a shift from a centralized to a distributed architecture. In order to conduct our research, we are using mobile data. Mobile has access to a ton of data, thus we are utilizing the data in cloud storage for additional research. Given the size of the data we need to analyze, HDFS is where we store it. One of the fundamental components that protects information from unfavorable and unwanted data is security. Analysis of current work explains why HDFS lacks a security framework or mechanism to hide and secure information. This paper suggests a solution for encrypting Big Data so that it can be safely stored in HDFS (Dede et al., 2016; Chen et al., 2016; Hajeer & Dasgupta, 2019).

The main goal of this work is to develop a method for securely sending and recovering data in a big data environment without increasing any overheads in computing, storage, or communication. A system of laws supports the methodology employed in the proposed work. The Salsa20 algorithm is suggested as a replacement for the AES method in this paper (Bernstein, 2018; Ding, 2019). Data privacy and protection from unauthorized usage are two basic requirements in the Hadoop framework. Information encryption is one of these requirements. Every encryption technique has the problem of added overhead, which should be reduced as much as practical. Salsa20 can replace the AES algorithm in existing work to improve the performance of the Hadoop environment during data encryption (Bai & Wu, 2016; Parmar et al., 2017).

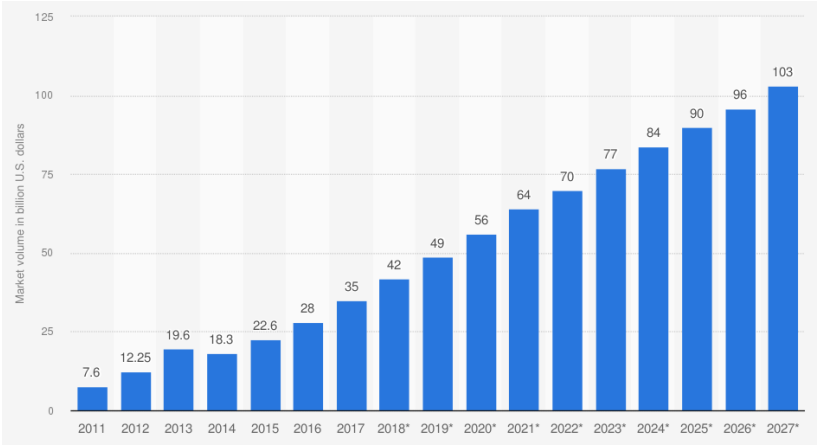

Fig. 1: Big data market size revenue forecast 2011 to 2027[1]

2. LITERATURE REVIEW

A distributed file system is the Hadoop Distributed File System (HDFS) (Dede et al., 2016; Chen et al., 2016). HDFS is designed as low-cost, highly fault-tolerant hardware. HDFS was thought to be more suited for applications that employ big data since it offers us great throughput when the massive data collection is used. A typical HDFS file might range in size from gigabytes to terabytes. As a result, HDFS is configured to support big files. It should scale to hundreds of nodes in a single cluster and offer high aggregate data bandwidth. Tens of millions of files should be supported at once.

Big data research usually involves Hadoop. It is expected that Hadoop would process vast amounts of data without caring too much about its organization. The MapReduce framework, which Google developed to solve the issue of web search indexes, is the fundamental component of Hadoop. A nonprofit organization called Apache Software Foundation (ACF) is in charge of managing and maintaining the Hadoop environment's technology as well as the framework. Ghemawat & Dean, 2008; Evermann, 2016) Hadoop employs the Map-Reduce programming architecture. Hadoop does not have any security mechanisms. It has been documented in a few investigations that data is encrypted before being stored in HDFS using cryptographic computations. To protect sensitive data, encryption is used (Song et al., 2017).

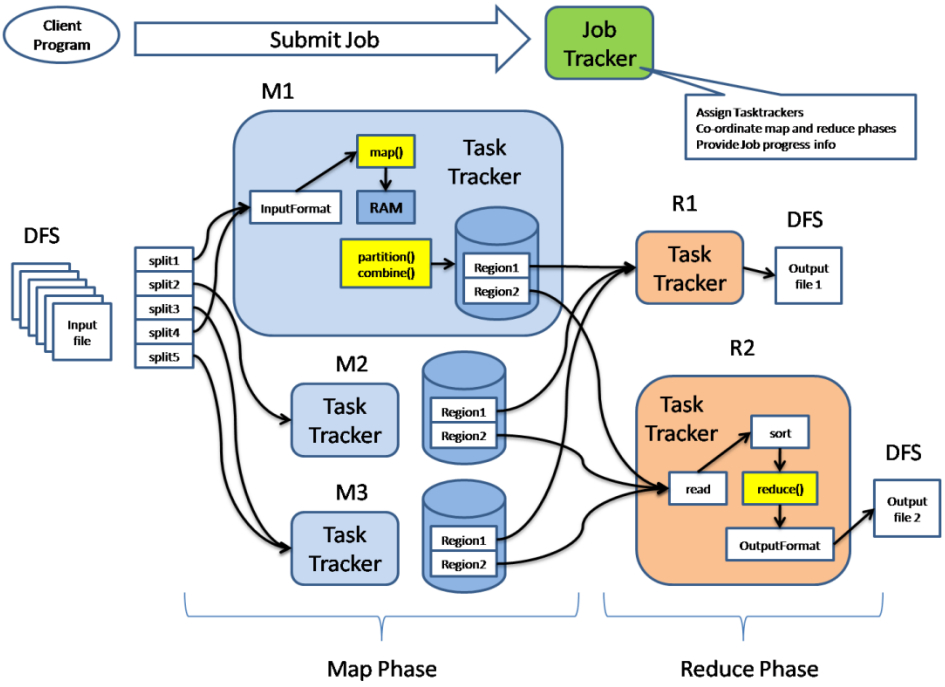

Fig. 2: Architecture of MapReduce in Hadoop

Through a series of substitutions and alterations, the initial message or piece of data is converted into ciphertext, which is then converted into an arbitrary message during the encryption process. Numerous advancements have been created in the Hadoop environment to enable individuals to perfect their own talents and experiment with Massive Information. To handle the enormous number of datasets effectively in this manner, many relationships and associations have a firm grasp of the MapReduce architecture, which tackles the problem with information that is overly directive.

The most crucial security and protection measures are those pertaining to big data, which are always stored in a shared storage place called HDFS. Thus, overcoming these obstacles and obtaining data in the most efficient manner is the primary requirement for the creation of Big Data analysis tools. Data is jumbled before being sent to the cloud to lower the possibility of sensitive information being lost. Converting text that is not encrypted into ciphertext is known as encryption, while the opposite is known as decryption.

In data security applications, multiple encryption calculations are frequently used. Using asymmetric (public) and symmetric (private) key encryption, you may group them accordingly. A single key is utilized for both data encoding and decoding in symmetric key and secret key encryption. There are two types of asymmetric keys: public and private. We looked at several symmetric and symmetric cryptography approaches, as well as recognized weaknesses, countermeasures, and comparisons (Wei et al, 2018; Lee, 2020; Gai et al., 2017). Lee et al. examine the safety of data in cloud computing with AES under the Heroku cloud (Lee et al., 2018).

Table 1: Comparison of Encryption Algorithms

Factor	DES	AES	RSA
Key size	56 bits	128, 192, 256 bits	>1024 bits
Ciphering & deciphering key	same	same	different
Algorithm	symmetric algorithm	symmetric algorithm	asymmetric algorithm
Encryption / Decryption performance	moderate	faster	slower
Power Consumption	low	low	high
Security	excellent	not secure	least secure
Ciphering / Deciphering	different	different	same

3. RESEARCH METHODOLOGY

The Salsa20 stream cipher divides a 256-bit key across 264 streams with 264 randomly accessible 64-byte blocks in each stream [7,8]. Salsa20 interprets the 64-byte input x as 16 words (0, 1, 2,..., 15) in little-endian form (0, 1,..., 232-1). 320 invertible adjustments, each changing one word, are applied to these 16 words. The final 16 words are added to the initial x0, x1, x2,..., and x15 words, respectively, modulo 232, to create the 64-byte output Salsa20(x) in little-endian form. Each adjustment entails rotating the sum of two more words modulo 232 and xoring it into one word. Consequently, the 320 alterations consist of 320 adds, 320 xors, and 320 rotations altogether. All of the rotations are done at fixed distances. Ten identical double-rounds make up the whole series of adjustments. There are two rounds in each double-round. There are four parallel quarter-rounds in each round. Four words are modified by each quarter-round. Figure 3 shows a structure of SALSA20 algorithms.

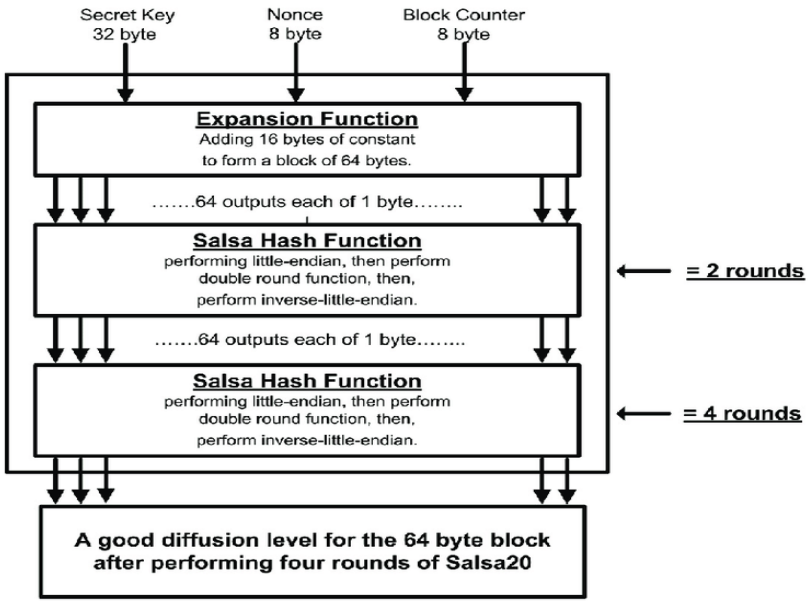

Fig. 3: Block Diagram of SALSA20

Using the mathematical function, we can get the 64-byte the output, which functions as a

keystream. Platform management, authentication, and encryption instruments greatly improve Hadoop cluster security by blocking even the most basic techniques used by attackers to pilfer data or jeopardize its value. Encryption ensures that there are two ways to get around application security measures. Encryption is also necessary to satisfy information management or consistency standards. In the first research, we focused much of our attention on integrating 10 Hadoop modules, each with a unique configuration and hidden inside the intricacy of the cluster, exposing each to a distinct attack surface from possible attackers.

Though Apache Ranger has a strong administrative plane for setting up settings and implementing data protection measures within the cluster, our primary concern is deployment validation. A scalable cryptographic key management system, the Ranger Key Management Service (Ranger KMS), is offered for HDFS "data at rest" encryption. Expanding upon Hadoop KMS, which was initially created by the Apache group, Ranger KMS allows framework administrators to store data in a secure database, hence increasing local Hadoop KMS capacity.

4. RESULTS

An experiment was conducted with a 10-node Hadoop cluster. The first eight nodes served as stores (as DataNode servers) and computation (as MapReduce clients) resources, while the ninth and 10th nodes oversaw NameNode storage and scheduled MapReduce, respectively. Initially, an 8-core processor, 8GB of RAM, including a Gigabit Ethernet NIC were installed on each of the eight nodes. Every node was running Java 7 and the Hadoop framework 3.2.4. 32GB RAM and the last two-node, 16-core CPU. Along with the operating system, the Hadoop database the application, and the application's scratch space, only HDFS data was stored on disk.

Hadoop's replication was switched off. stages involved in sending mobile data packets (pushed to a cluster of 10 nodes for HDFS storage) and decrypting them.

Information security has made use of a number of cryptographic methods that are available. There are several types of algorithms: symmetric-key algorithms include Triple DES, Advanced Encryption Standard (AES), and Data Encryption Standard (DES). Asymmetric key algorithms include RSA and Elliptic Curve Diffie-Hellman (ECDH). Because they are often employed in recent articles, we have included AES and Blowfish approaches for comparison. Several data sizes, ranging from 1MB to 10GB, were used to compare AES and BlowFish with our proposed methodology. The encryption timings for each scheme are displayed in Figure 4.

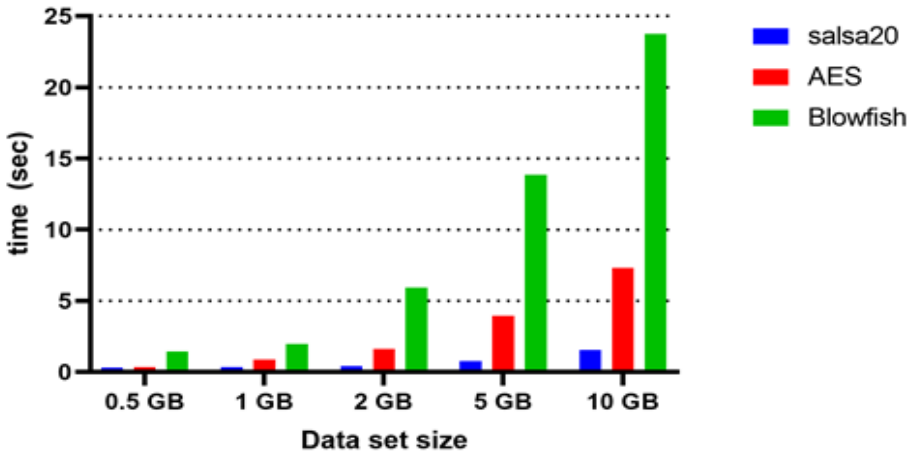

Fig. 4: Encryption time of algorithms

Every encryption method has its advantages and disadvantages. To apply a good cryptography computation to our application, we need to grasp how the different algorithms work as well as their benefits and drawbacks. This paper has evaluated the following measures, based on which the cryptosystems could prove most helpful for our goal. The estimated encryption time, expressed in seconds, comes first. The encryption time has an effect on how the framework works. Encryption requires less time to complete, making the foundation responsive and fast. For the framework to function more quickly and responsively, the second parameter, Decryption Time, needs to be less than Encryption Time. For our test, the decryption time was measured in seconds.

The third and most important parameter is the avalanche effect, which makes a little modification to the input have a significant effect on the message that's generated. A good algorithm has a high avalanche impact. As a result, a more detailed representation of the three factors and their effects on our experiment is provided below. The duration of each approach for different data amounts is shown in Figure 2. The results of the investigation show that even bigger data sets may be encrypted faster using our recommended method. Based on the data, Salsa20 seems to be the best choice for our application.

Block ciphers like Salsa20 can be used since it was designed to be capable to encrypt any random blocks of data. A quality that is highly appreciated about this approach is its capacity to decode each encrypted block independently, given the key and nonce needed for the decryption. Figure 5 illustrates the decoding time. Therefore, encrypted information stored in HDFS may be directly used to conduct MapReduce jobs.

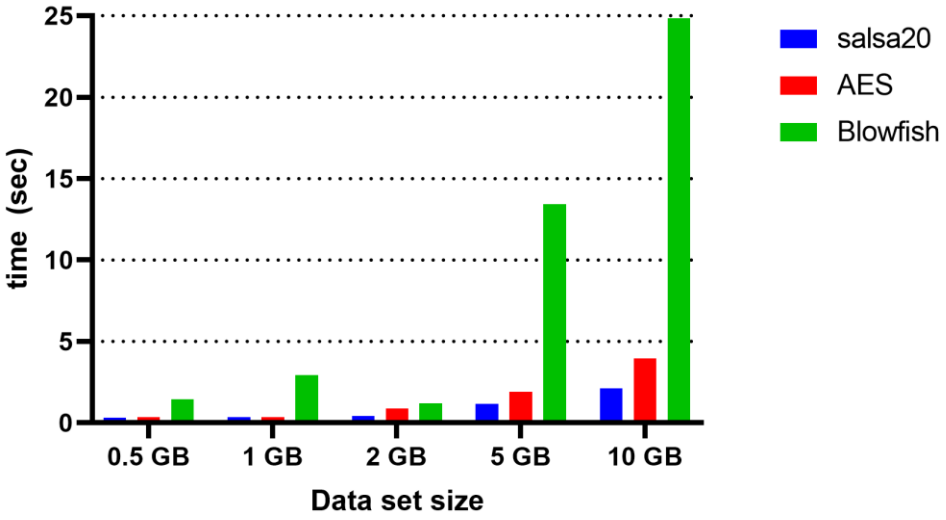

Fig. 5: Decryption time of algorithms

AES decryption provides a higher throughput as compared with encryption. For a few reasons, however, it appears differently whereas encrypting and decrypting many blocks. For example, if you use cipher-block chaining (CBC), you have to encrypt each block in turn (encrypt block 0 first, then block 1, then block 2, etc.). However, decryption can proceed simultaneously because the XOR step (using previous blocks of ciphertext) is carried out after the block cipher is applied. It is possible to determine the throughput of the encryption/decryption method by multiplying the total amount of plain texts in megabytes by the total length of time needed for encryption as well as decryption. Consequently, our technique was regarded as having a high speed as the algorithm's throughput rose.

Salsa20 is faster at encrypting and decryption, as seen by Figure 6's algorithm throughput. This unequivocally demonstrates that Salsa20 is the optimal algorithm for our research.

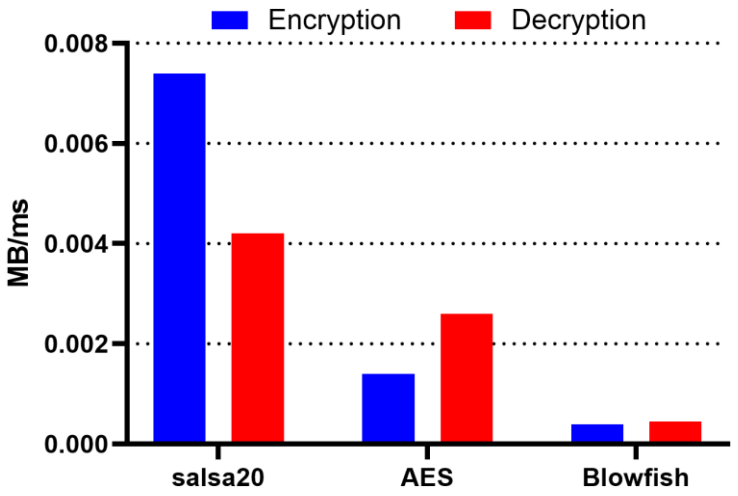

Fig. 6: Encryption/Decryption time of algorithms

In a security analysis, we evaluated the suggested algorithm using the Avalanche Effect security metric. It should be noted that Avalanche Effect's main benefit is its ability to measure the strength of the suggested calculation against hacking and breaking risks as well as persistent attacks, such as brute force assaults. A few bits in the original message can be modified in key encryption techniques, causing an avalanche of changes that result in numerous content piece changes. The difference in one plaintext bit must yield one-half of the ciphertext for an algorithm to satisfy the avalanche requirement [35]. To check the Avalanche effect, the formula was written as follows.

Rate of avalanche effect = size of flipping bits in ciphertext/size of bits in ciphertext × 100

Table 2 displays the outcomes of calculating each unique Avalanche Effect. Figure 7 illustrates the effect of the avalanche on benchmark encryption calculations, including Salsa20. It can be demonstrated that only AES and Salsa20 encryptions meet the avalanche impact criteria; hence, their chances of breaking are higher. Because of this, Salsa20 is significantly more practical and reliable than other benchmark algorithms.

Based on experiments, it has been shown that the suggested framework with Salsa20 has the biggest avalanche effect, the lowest run time, and the maximum throughput. Consequently, the novel structure that has been proposed is scalable, fast, safe, reliable, and dependable. The paper's suggested structure offers functionality, scalability, or performance.

Table 1: Rate of avalanche effect for encryption algorithms

Encryption algorithm	Count of flipped bits	Rate
DES	28	24.1
3-DES	26	30.5
AES	58	52.3
MARS	30	49.3
Blowfish	64	48.3
Salsa20	72	56.7

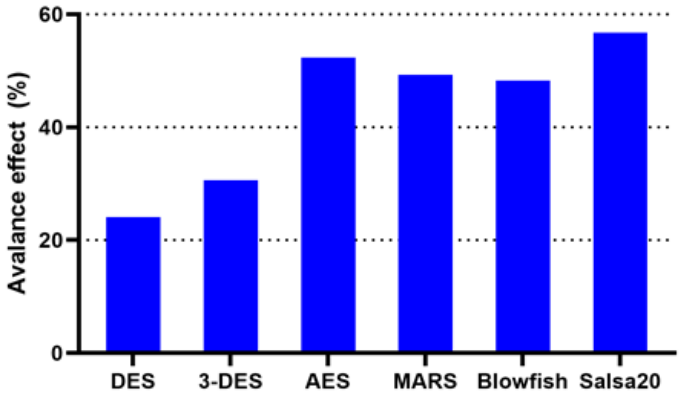

Fig. 7: Comparison of avalanche effects by algorithm

5. CONCLUSION

Information encryption techniques over plain text are a must for the Hadoop system. The Salsa20 encryption method and Ranger Key Management Service (Ranger KMS) would be used in the suggested approach. The features, constraints, and current research directions in the large security of data in the cloud context are distinguished in this study. The multi-node Hadoop cluster's transmission speed and computation time will serve as the foundation for estimating the overall labor cost. According to the endeavor's ultimate conclusion, the suggested solution ensures control over access, the authentication process, and confidentiality on the Hadoop server. Before mining, it evaluates the client's access rights and privileges. By following this process, user information is protected from that of the legitimate data owner, superfluous mining effort is eliminated, and permitted user requests are ensured. A Hadoop cluster with 10 nodes allows for quick computing. As a result, as the new method that has been suggested shows, this configuration might be used with Hadoop to maintain security in HDFS for important data, particularly for the cloud mobile storage of information that offers safety with low computational cost.

ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2021R1F1A1052129)

References

AlZubi A. A. (2023). Application of Machine Learning in Drone Technology for Tracking Cattle Movement . Indian Journal of Animal Research. 57(12): 1717-1724. https://doi.org/10.18805/IJAR.BF-1697 .

Bai, K. & Wu, C. (2016). An AES-Like Cipher and Its White-Box Implementation. The Computer Journal, 59(7), 1054-1065. https://doi.org/10.1093/comjnl/bxv119.

Bernstein, D. J. (2018). ‘The Salsa20 family of stream ciphers. New Stream Cipher Designs. Lecture Notes in Computer Science, 4986, 84–97. https://doi.org/10.1007/978-3-540-68351-3_8

Chen, D., Chen, Y., Brownlow, B. N., Kanjamala, P. P., Arredondo, C. A. G., Radspinner, B. L. & Raveling, M. A. (2016). Real-Time or Near Real-Time Persisting Daily Healthcare Data Into

HDFS and ElasticSearch Index Inside a Big Data Platform. IEEE Transactions on Industrial Informatics, 13(2), 595-606. https://doi.org/10.1109/TII.2016.2645606.

Cho, O.H., Na, I.S. and Koh, J.G. (2024). Exploring Advanced Machine Learning Techniques for Swift Legume Disease Detection. Legume Research. https://doi.org/10.18805/LRF-789

Dede, E., Sendir, B., Kuzlu, P., Weachock, J., Govindaraju, M. & Ramakrishnan, L. (2016). Processing Cassandra Datasets with Hadoop-Streaming Based Approaches. IEEE Transactions on Services Computing, 9(1), 46-58. https://doi.org/ 10.1109/TSC.2015.2444838.

Ding, L. (2019). Improved Related-Cipher Attack on Salsa20 Stream Cipher. IEEE Access, 7, 30197-30202. https://doi.org/10.1109/ACCESS.2019.2892647.

Evermann, J. (2016). Scalable Process Discovery Using Map-Reduce. IEEE Transactions on Services Computing, 9(3), 469-481. https://doi.org/10.1109/TSC.2014.2367525

Gai, K., Qiu, M. & Zhao, H. (2017). Privacy-Preserving Data Encryption Strategy for Big Data in Mobile Cloud Computing. IEEE Transactions on Big Data, 7(4), 678-688, https://doi.org/10.1109/TBDATA.2017.2705807.

Ghemawat, S. & Dean, J. (2008). MapReduce: Simplified data processing on large clusters. Communications of the ACM, 51(1), 107-113. https://doi.org/10.1145/1327452.1327492

Hajeer, M., & Dasgupta, D. (2019). Handling Big Data Using a Data-Aware HDFS and Evolutionary Clustering Technique. IEEE Transactions on Big Data, 5(2), 134-147. https://doi.org/10.1109/TBDATA.2017.2782785.

Lee, B. H., Dewi, E. K. & Wajdi. M. F. (2018). Data security in cloud computing using AES under HEROKU cloud. 27th Wireless and Optical Communication Conference (WOCC), 1-5. https://doi.org/10.1109/WOCC.2018.8372705

Lee, K. (2020). Comments on "Secure Data Sharing in Cloud Computing Using Revocable-Storage Identity-Based Encryption. IEEE Transactions on Cloud Computing, 8(4), 1299-1300, https://doi.org/10.1109/TCC.2020.2973623

Park, J. K. (2022). Machine Learning Algorithm to Predict Wildfire Occurrence Based on IoT Sensor Data. Journal of Next-generation Convergence Technology Association, 6(6), 967-973. https://doi.org/10.33097/JNCTA.2022.06.06.967

Parmar, R. R., Roy, S., Bhattacharyya, D., Bandyopadhyay, S. K. & Kim, T. (2017). Large-Scale Encryption in the Hadoop Environment: Challenges and Solutions. IEEE Access, 5, 7156-7163. https://doi.org/10.1109/ACCESS.2017.2700228

Sandhu, A. K. (2022). Big data with cloud computing: Discussions and challenges. Big Data Mining and Analytics, 5(1), 32-40. https://doi.org/10.26599/BDMA.2021.9020016

Song, Y., Shin, Y., Jang, M. & Chang, J. (2017). Design and implementation of HDFS data encryption scheme using ARIA algorithm on Hadoop. IEEE International Conference on Big Data and Smart Computing (BigComp), 84-90. https://doi.org/10.1109/BIGCOMP.2017.7881720

Taylor, P. (2022). Big data market size revenue forecast worldwide from 2011 to 2027. Statista. https://www.statista.com/statistics/254266/global-big-data-market-forecast/.

Vainshnav, S. P., & Dave, K. K. (2022). A study on Artificial Intelligence and Machine Learning in Banking Sector with special reference to term loan. Pacific Business Review (International). 15(3), 34-53.

Wei, J., Liu, W. & Hu, X. (2018). Secure Data Sharing in Cloud Computing Using Revocable-Storage Identity-Based Encryption. IEEE Transactions on Cloud Computing, 6(4), 1136-1148. https://doi.org/10.1109/TCC.2016.2545668