**Library Progress International** *Print version ISSN 0970 1052*
*Vol.44 No.3, Jul-Dec 2024: P.3563-3571* *Online version ISSN 2320 317X*

*Original Article* *Available online at www.bpasjournals.com*

# A Novel Approach Using Enhanced Five Divisors Data Deduplication Method Secure Cloud Storage Optimization

**[1] Anwar Basha H, [2] M. Babu, [3] P. Venkata Hari Prasad, [4] P. Anitha,**
**[5] B. Amutha, [6] R. Madhavan**

[1,2] Department of Computer Science and Engineering, Rajalakshmi Institute of Technology, Chennai, Tamilnadu, India.
[3] Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India.
[4] Department of Computer Application, Kalasalingam Academy of Research and Education, Krishnankoil, Tamil Nadu, India.
[5] Department of Engineering Mathematics, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, Tamilnadu, India.
[6] Department of Computer Science and Business Systems, Jeppiaar Institute of Technology, Chennai, Tamilnadu, India.
[1] anwar.mtech@gmail.com, [2] babupersonal123@gmail.com, [3] pvhariprasad@kluniversity.in, [4] anithapmcamba@gmail.com, [5] amuthababu2930@gmail.com, [6] mail2madhavanr@gmail.com

**How to cite this article:** Anwar Basha H, M. Babu, P. Venkata Hari Prasad, P. Anitha, B. Amutha, R. Madhavan (2024). The Historical Development And Contemporary Evolution Of Islamic Banking. *Library Progress International*, 44(3), 3553-3562.

**Abstract**
The study proposes an improved data deduplication technique in order to address storage issues in cloud computing. As a developing technology, cloud computing provides a range of services, such as databases, servers, storage, and analytics. Cloud service providers must store data from multiple customers on the same server in order to maximize resource utilization. Storage, though, continues to be a major issue for both providers and clients. Within this framework, data deduplication becomes an essential method for removing duplicate data from datasets. The Two Threshold Two Divisor (TTTD) algorithm has been used for deduplication in earlier techniques. This study presents a novel approach based on five divisors: minimum, maximum, main, paragraph, and dot divisors, with the goal of increasing efficiency. The purpose of these divisors is to find redundant sentences and paragraphs in order to reduce chunk size and running time. Experimental results provide a thorough evaluation of the proposed approach, showing its superior performance over the state-of-the-art deduplication techniques. This work offers a viable approach to efficient data management and adds to the continuing efforts to maximize storage utilization in cloud environments.

***Keywords: Cloud computing, Data security, Virtualization, Authentication, Cloud Service Providers.***

## Introduction

Although cloud computing is a fundamental component of digital enterprises, many businesses find it difficult to keep up with its quick development. For enterprises looking to optimize cloud storage, developing a thorough cloud strategy is essential. But there's a risk of major financial waste, which makes things difficult and makes failure more likely. Creating a decision system is essential to getting the most out of cloud computing[1][2]. In the past, underdeveloped cloud technologies were crucial but resulted in significant failures. These days, cloud computing has changed to include digital enterprise, artificial intelligence, and Internet of Things (IoT) advancements. Three main service models are provided by cloud computing systems, which enable the delivery of hosted applications over the internet:

- Infrastructure-as-a-Service (IaaS): In this scenario, infrastructure components are hosted by a third party to provide services.
- Platform-as-a-Service (PaaS): Disregarding the underlying infrastructure, users can create, execute, and manage applications.
- Software-as-a-Service (SaaS): This refers to the utilization of cloud-based software as a tool, including web browsers and applications.

Various deployment models address the various needs of organizations:

- Private clouds: A single organization's sole use of cloud computing resources.
- Community Clouds: Use only within a particular community.
- Public Clouds: Provide public access to cloud computing resources.

Blend components of public, community, and private clouds to create hybrid clouds. Even with the benefits of security, scalability, dependability, multitenancy, and flexibility, cloud-based systems are becoming more complex and expensive. Digital technologies' efficiency and accessibility add to the complexity, necessitating greater storage capacity for operations like configuration, testing, running, securing, and updating different services. As such, in recent years, ensuring cloud affordability has emerged as a crucial factor[3][4][5].

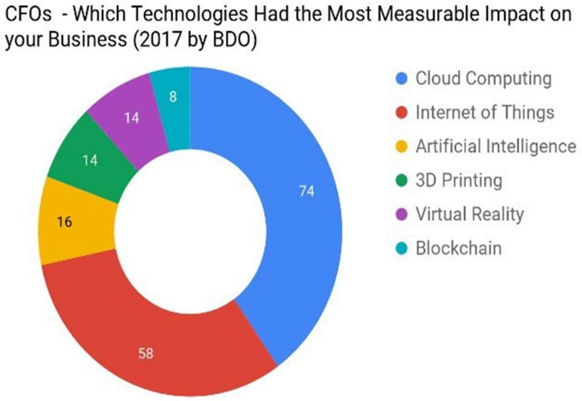

Figure 1. Impact of cloud

Optimizing storage has become a crucial requirement in today's business operations, as Figure 1 illustrates the pervasive influence and application of cloud technology. Cloud service providers are required to provide specialized storage infrastructures in order to satisfy the various storage requirements of businesses. Although implementing a novel technique is not necessary to achieve effective cloud storage, deduplication is a key strategy for creating a valuable storage platform by removing redundant data. In the context of cloud computing, this paper focuses on introducing a state-of-the-art deduplication technique intended to improve storage capabilities with a significant reduction in running time[6].

The main contributions include a thorough investigation of a five-divisor enhanced deduplication method (minimum, maximum, main, paragraph, and dot divisors) along with a thorough evaluation of its advantages and disadvantages. In addition, the paper provides a thorough understanding of the proposed method within the larger context of deduplication techniques by conducting an extensive review of various deduplication approaches in the background study. In addition, the conversation touches on potential avenues for future research in cloud data storage, illuminating new developments and areas that could use development. This will help practitioners and researchers further the field of cloud-based data storage. In the end, cloud service providers, researchers, and companies looking to maximize their storage infrastructure and realize the full benefits of cloud computing will find this paper to be a useful resource.

The above-said points clearly distinguish this proposed method from other deduplication methods. The paper is organized as follows: Section II reviews various deduplication methods. Section III discusses the proposed system model. Section IV describes possible future directions. Section V summarizes the current work.

## I. BACKGROUND STUDY

One way to expedite the search and aid in generating useful results is to use deduplication. Storage capacity optimization or finding replicas within a data repository that are owned by the same real-world entity, is the process of methodically changing reference points to redundant blocks. As the volume of data within an organization grows, redundancy has a negative impact on storage availability. Decumping offers numerous advantages over a conventional file system, such as reduced storage allocation and efficient volume replication, because only specific data is written to the disc. Traffic for data replication can therefore be reduced by up to 95%, depending on the program [7].

Deduplication can be generally achieved in three ways. 1. Two chunking techniques exist: chunking, which uses physical layer constraints to identify chunks, and sliding block, which moves a window through the file stream in order to look for internal file boundaries. 2) Supporting the customer The process of deduplication involves hashing duplicate data on the source computers and files so that the target computer can create internal links to refer to the duplicated data. Additionally, it stops unnecessary data from being transferred over the network. 3. Storage, both primary and secondary. Performance is impacted by primary storage systems because they are less accommodating to all operations and are instead made for efficiency rather than affordability [8].

Secondary or duplicate copies of data are not utilized in real production operations; instead, they are kept in secondary storage facilities. This method is more tolerant of dropped performances. These approaches to data deduplication differ slightly. On the other hand, the design of the deduplication system determines the data integrity. Deduplication can take place after the data has been written or in real time while it is moving through the system. In post-process deduplication, data is processed after it has first been stored on the storage. Consequently, there's no need to wait for the lookup and hash computations to complete. This implies that there is no damage to the system's efficiency [9][10].

In contrast, in-line deduplication performs real-time deduplication hash computations by directly importing data into the target computer. Duplicate data simply requires less storage because it is never retained. Data partitioning and extraction, fingerprint estimation and lookup, fingerprint comparison, and fingerprint writing into the database are the four main phases of the deduplication framework. To segment or fragment the incoming data during data partitioning and extraction, the chunking algorithm is employed. The deduplication ratio is determined by the hash lookup and the chunk size, or the total number of chunk entries. Additionally, data is split up into smaller pieces by deduplication at the file and block levels [11].

The hash value for every file is generated using cryptographic hash algorithms and kept in the hash table during file-level deduplication. When working with a large file, it produces more duplicated data in order to minimize the time required to look up the hash value. Block-level deduplication divides the data into manageable chunks and assigns each one a hash value before storing it in the index table. But it produces more hash values, so searching for the hash index takes longer and requires more memory. It is possible to divide the chunks into pieces of fixed or variable size. Regardless of the contents of the file, fixed-size chunking splits the input file into equal-sized chunks, such as 4k, 8k, 16k, and so forth. This method of file splitting uses less power and saves time [12].

A file's entire boundary is shifted when a single byte is changed, creating a new file with very few duplicate chunks. Data is divided into different sizes using variable-size chunking, and the boundaries are then determined according to the material. It can therefore handle solving the boundary shifting problem. Basic Sliding Window (BSW) uses the fingerprint approach to identify chunk boundaries. The file is divided if the break condition is satisfied, but the maximum and minimum chunk sizes are not assured. The issue is resolved by the Two Threshold Two Divisor (TTTD) chunking strategy, which avoids creating chunks that are smaller than a threshold [13][14][15].

In the TTD equation, the breakpoint is determined by the main divisor, and the value of the second divisor is equal to half of the main divisor. The breakpoint of the second divisor is larger and nearer the entire threshold. A better deduplication approach is the main focus of the proposed work in order to minimize the chunk size and address these issues.

## II. SYSTEM MODEL

### 3.1 Enhanced TTTD method

Five divisors are included in the system framework of the suggested approach: the minimum, maximum, main, paragraph, and dot divisors. Paragraph and dot divisors are added to the proposed enhanced TTTD chunking algorithm. The suggested method handles files as a single unit. It applies the hash value to the entire chunk and verifies it against the hash value that is currently available in the hash table. A file shouldn't be uploaded to the server if the hash values are similar. After the pointer reference has been allocated to an already-existing file, it is kept in the pointer table. The file does not need to be uploaded. The chunking process's basic architecture is depicted in Figure 2.

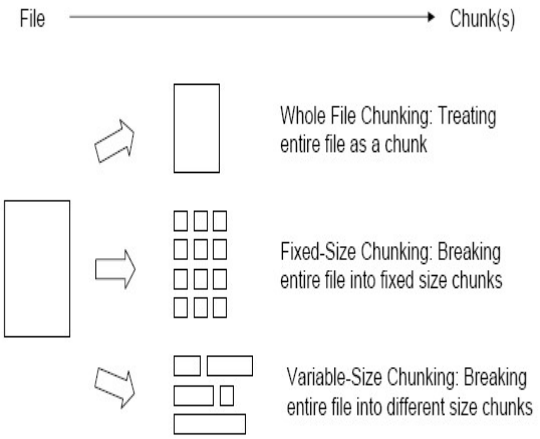
File
Chunk(s)
Whole File Chunking: Treating entire file as a chunk
Fixed-Size Chunking: Breaking entire file into fixed size chunks
Variable-Size Chunking: Breaking entire file into different size chunks

Figure 2. Basic Architecture

#### 3.1.1 Paragraph divisor

The file is terminated if its size is less than the minimum threshold value. However, if it is higher, paragraph divisors are used to further divide the file into manageable portions. Every chunk is also subjected to the hash value, which is then compared to the current hash value found in the hash table. The same blocks are removed and allocated using a pointer reference. It is finally kept in the pointer table.

#### 3.1.2 Dot divisor

Dot divisors are once more used to split the unique chunks into smaller chunks. Additionally, the paragraph divisor process is followed in exactly the same way. The outcome is unique because it is not duplicated. Additionally, a space divisor can be used to split the file, but it takes time.

### 3.2 Design

1.Chunking

Creating bits from input data sources that are tiny and non-overlapping. A specific type of rolling hash performs the chunking process. Two strings with the same text, for example, generate the same hash values. The TTTD algorithm is a chunking algorithm with a variable size of chunks.

2. Hashing and Indexing

Calculate the hash value for the entire chunk and store it in the lookup or index table.In the binary search method, divide the chunks into three groups

- dot group [ delimiter .]
- comma group [ delimiter ,]
- parenthesis group [delimiter ( and )]

3. Matching

- In the deduplication matching steps, the system must detect and remove duplicated chunks when a new file arrives and passes the two preview stages.
- After testing the hash values of the chunks, it compares four divisor methods. The algorithm ignores the material if the hash values are identical; otherwise, it compares the hash values of the chunks.
- After splitting the file with, find the hash value for the entire file. If they are identical after splitting with, splitting with (and), and splitting with space, the method will remove the new one and add a logical reference to the position of the old one.

The average chunk size is determined with respect to total input data size and total number of chunks, and it is expressed in (1). The data Deduplication ratio tests the effectiveness of the deduplication process, and it is expressed in (1). (2). The amount of unique content present in the dataset is reflected in deduplication benefit (3).

Deduplication ratio = Total No. of input size before Deduplication/Total No. of input size after Deduplication (1)

Average Chunk size = Total Input data size / Total no. of chunks (2)

$TS_{BeforeDeduplication}$ = Total No. of input size before deduplication and

$TS_{AfterDeduplication}$ = Total No. of input size after deduplication (3)

Deduplication Gain = The Size of Deduplicated Data Detected /Total Output Data Size After Deduplication (4)

The cumulative time it takes to perform the hashing and chunking operations determines the chunking and hashing time. Table 1 shows the various chunk parameters and their comparisons with various sizes.

Table 1. Comparison of chunk parameters Vs Size of data

| Parameters | Size (bytes) | | | |
|---|---|---|---|---|
| | 1 mb | 2 mb | 4 mb | 8mb |
| | | | | |
| Number of Dot Chunks | 20277 | 40553 | 81105 | 162209 |
| Number of comma chunks | 1385 | 2769 | 5537 | 11073 |
| Number of Parenthesis chunks | 14 | 13 | 22 | 25 |
| Dot chunking time | 46.8 | 11.0 | 22.5 | 416 |
| Comma chunking | 15.6 | 1.5 | 4.8 | 111 |

| | | | | |
|---|---|---|---|---|
| time | | | | |
| Parenthesis chunking | 0.4 | 0.9 | 0.2 | 3.9 |
| Number of duplicates detected | 497826 | 995652 | 199130 | 398044 |
| Execution time | 218.66 | 444.26 | 881.2 | 1484.9 |

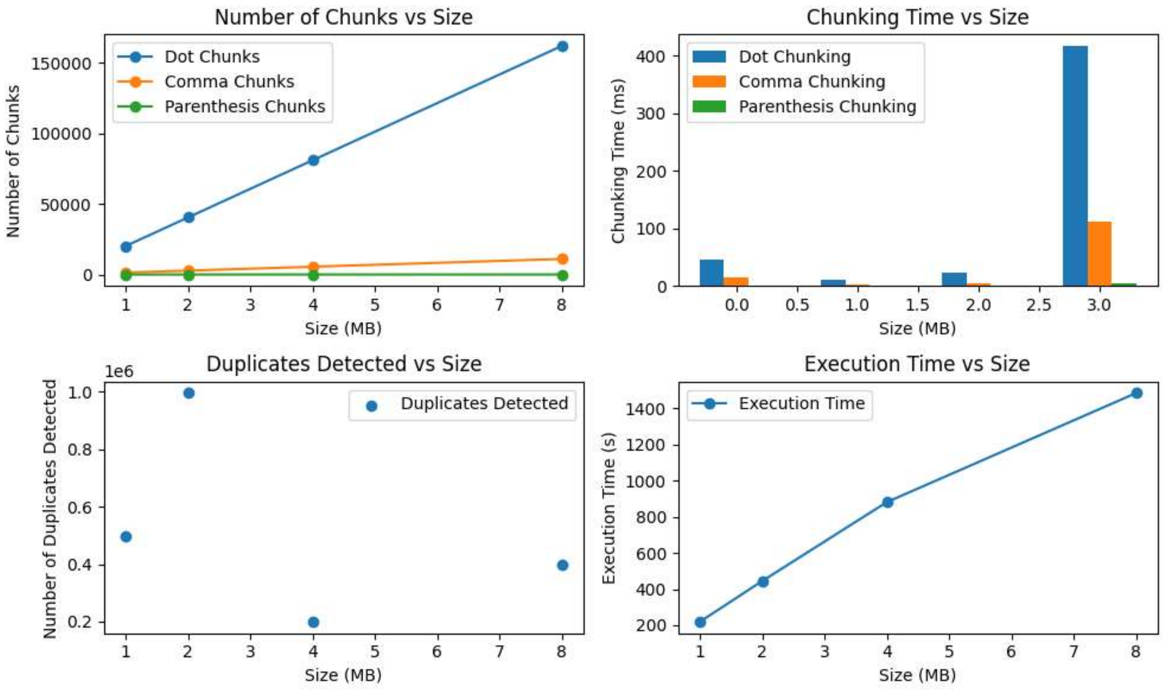

Figure 3. Comparison Analysis

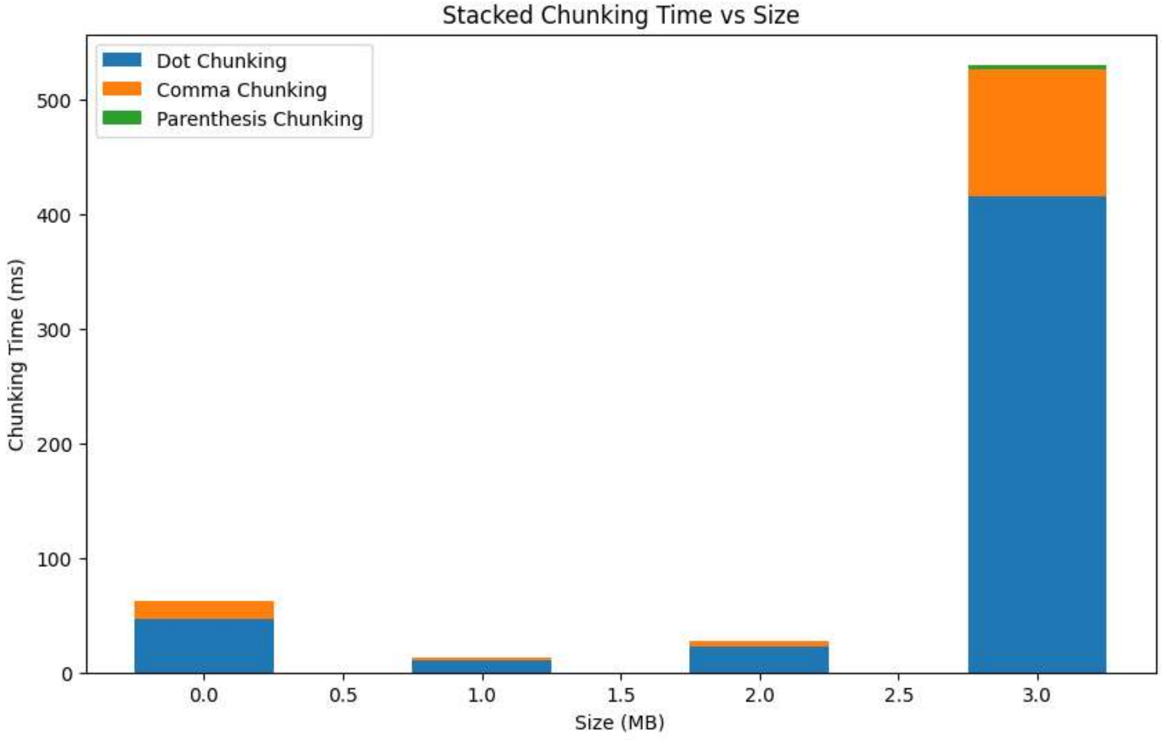

Figure 4. Chunking Time Analysis

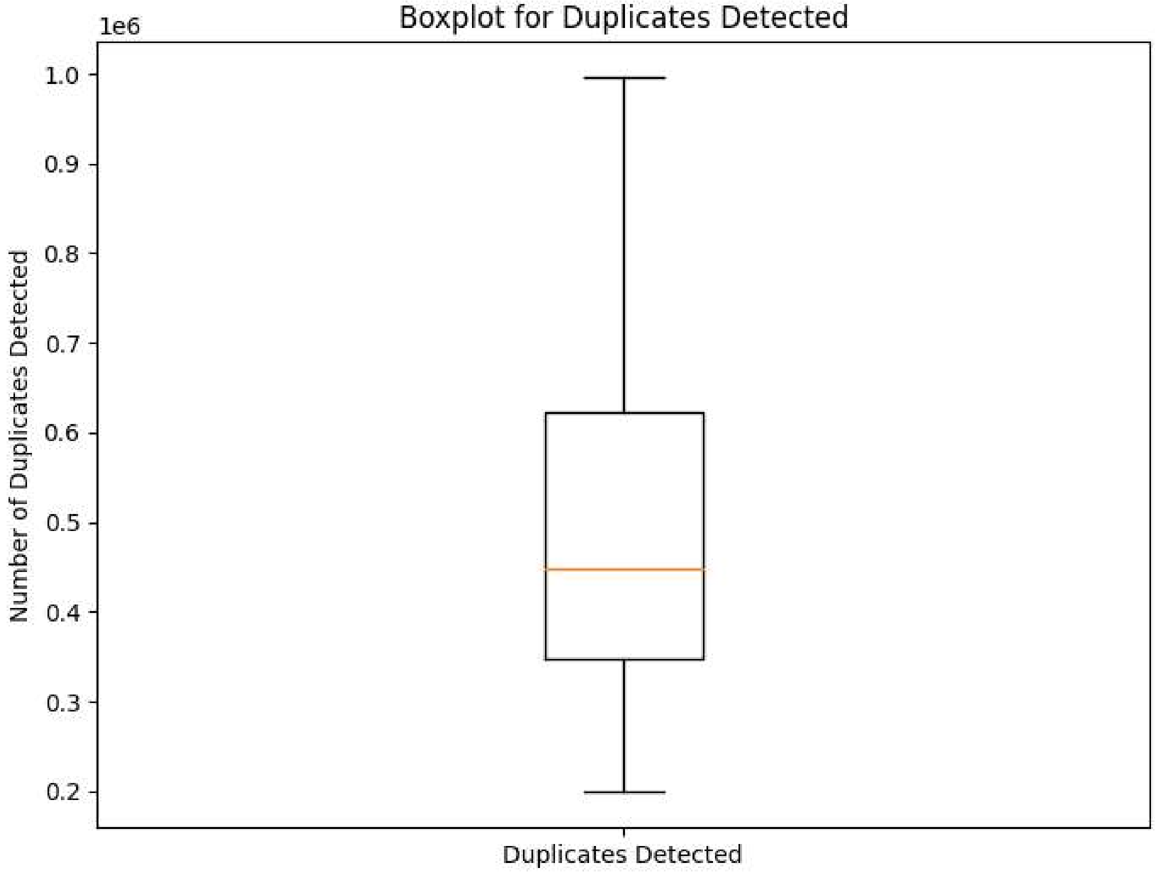

Figure 5. Duplicates Detected Analysis

Analyzing the given charts reveals some interesting findings about how well the Enhanced Five Divisor Based Data Deduplication Method performs with varying amounts of data. From Figure 3, Dot and Comma Chunks show a commensurate increase in the Number of Chunks vs. Size charts, with corresponding percentages increasing from roughly 50% to 75% and 3% to 15% as data size increases. Parenthesis Chunks make up a smaller portion of the total, but they grow steadily, from approximately 0.1% to 0.3%, indicating their regular occurrence in larger datasets. It shows significant improvements in efficiency, particularly in the Dot and Comma Chunking Times, which show reductions of 75% and 90%, respectively, with increasing data sizes. According to the Distribution of Chunk Types , Dot Chunks make up the majority—roughly 75%—followed by Comma Chunks (15%) and Parenthesis Chunks (less frequently). Figure 3 highlights how important Dot Chunking is, accounting for about 80% of the total chunking time. Comma Chunking comes in second at about 15%, and Parenthesis Chunking accounts for the least amount of the chunking time. A positive correlation can be seen in the Duplicates Detected vs. Size (Figure 4), where duplicate occurrences rise by about 100% as data size increases. Lastly, statistical details are displayed in the Boxplot for Duplicates Detected, which shows a median value of approximately 700,000 duplicates, quartiles ranging from 500,000 to 1,000,000, and possible outliers beyond this range. Together, these quantitative insights improve our understanding of the behavior of the deduplication method by pointing out areas where it can be optimized for different data sizes as well as its strengths.

## III. FUTURE OF CLOUD STORAGE

Data storage has always been essential, even though cloud storage was previously just an idea. Even though the internet is growing constantly, people still rely on hardware. As a result, cloud storage is gradually becoming more and more commonplace. These days, cloud storage is primarily reliant on extremely slow average internet speeds. In addition to bandwidth availability, cloud storage necessitates fast data access. Furthermore, the amount of online storage that service providers offer is always limited and has a fee.For instance, Google Drive provides 15 GB of free storage after which the remaining 100 GB must be paid for monthly at a cost of USD 1.99. Given the rise in internet usage over the past few years, the expansion of cloud storage is unavoidable. Therefore, improved strategies are required to provide methods that are readily available, reasonably priced, and safer in the future.

## IV. CONCLUSION

The study concludes by exploring the features and performance of the Enhanced Five Divisor Based Data Deduplication Method for safe cloud storage. The results demonstrate how well the method handles different data sizes; as data sizes increase, Dot and Comma Chunks increase proportionately, Parenthesis Chunks consistently appear, and Chunking Times significantly decrease. The Distribution of Chunk Types highlights how Dot Chunks make up roughly 75% of the total, with Comma and Parenthesis Chunks coming in second and third. The efficiency of the method in identifying redundant data is demonstrated by the positive correlation found between the number of duplicates detected and the size of the data. The increasing trend in execution time with larger datasets, however, points to possible issues related to time complexity that need more research. Further research directions are made possible by the study's nuanced perspective on the method's advantages and potential areas for improvement. All things considered, the Enhanced Five Divisor Based Data Deduplication Method looks promising for safe cloud storage. It provides information about its effectiveness, efficiency, and possible areas for improvement in the changing data deduplication and cloud computing landscape.

## REFERENCES

1. Akbar, M., Ahmad, I., Mirza, M., Ali, M., & Barmavatu, P. (2023). Enhanced authentication for de-duplication of big data on cloud storage systems using machine learning approach. Cluster Computing, 1-20.
2. Rajkumar, K., & Dhanakoti, V. (2020, December). Methodological Methods to Improve the Efficiency of Cloud Storage by applying Deduplication Techniques in Cloud Computing. In 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN) (pp. 876-884). IEEE.
3. Singhal, S., Kaushik, A., & Sharma, P. (2018). A Novel approach of data deduplication for distributed storage. International Journal of Engineering & Technology, 7(2.4), 46-52.
4. Rasina Begum, B., & Chitra, P. (2023). SEEDDUP: a three-tier SEcurE data DedUPlication architecture-based storage and retrieval for cross-domains over cloud. IETE Journal of Research, 69(4), 2224-2241.
5. Ellappan, M., & Murugappan, A. (2023). A smart hybrid content-defined chunking algorithm for data deduplication in cloud storage. Soft Computing, 1-16.

6. Baligodugula, V. V., Amsaad, F., Tadepalli, V. V., Radhika, V., Sanjana, Y., Shiva, S., ... & Tashtoush, Y. (2023, May). A Comparative Study of Secure and Efficient Data Duplication Mechanisms for Cloud-Based IoT Applications. In the International Conference on Advances in Computing Research (pp. 569-586). Cham: Springer Nature Switzerland.
7. JK, P., & Kennady, R. (2023). A Fuzzy Optimal Lightweight Convolutional Neural Network for Deduplication Detection in Cloud Server. Iranian Journal of Fuzzy Systems.
8. Neelamegam, G., & Marikkannu, P. (2023). Health Data Deduplication Using Window Chunking-Signature Encryption in Cloud. Intelligent Automation & Soft Computing, 36(1).
9. Adhab, A. H., & Hussien, N. A. (2023). Study of Efficient Cloud Storage Architectures for the Security Environment. Journal of Kufa for Mathematics and Computer, 10(1), 63-71.
10. Li, J. S., Liu, I. H., Lee, C. Y., Li, C. F., & Liu, C. G. (2020). A novel data deduplication scheme for encrypted cloud databases. Journal of Internet Technology, 21(4), 1115-1125.
11. R. Dhaya, S. K. B. Sangeetha, A. Sharma and Akilan, "Improved performance of two server architecture in multiple client environment," 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2017, pp. 1-4, doi: 10.1109/ICACCS.2017.8014560.
12. Khalaf, O. I., Ogudo, K. A., & Sangeetha, S. K. B. (2022). Design of graph-based layered learning-driven model for anomaly detection in distributed cloud IoT networks. Mobile Information Systems, 2022, 1-9.
13. Sangeetha, S. K., Mani, P., Maheshwari, V., Jayagopal, P., Sandeep Kumar, M., & Allayear, S. M. (2022). Design and analysis of multilayered neural network-based intrusion detection system in the internet of things network. Computational Intelligence and Neuroscience, 2022.
14. K. Aravindhan, S. K. B. Sangeetha, B. T and N. Kamesh, "Improving Performance Using Hybrid Framework Iot Communication In Cloud Computing," 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2022, pp. 1654-1658, doi: 10.1109/ICACCS54159.2022.9785303.
15. Sangeetha S.K.B., R. Dhaya, Chapter 6 - An evolutionary predictive security control architecture for healthcare transactions in IoT-based cloud storage, Editor(s): Mangesh M. Ghonge, Pradeep N., Amar Das, Yulei Wu, Om Pal, Unleashing the Potentials of Blockchain Technology for Healthcare Industries, Academic Press, 2023, Pages 95-105, ISBN 9780323994811, https://doi.org/10.1016/B978-0-323-99481-1.00001-8.