SECURE DEDUPLICATION FOR CLOUD STORAGE USING MEMORY MAPPING TECHNIQUE FOR IMPROVING PERFORMANCE AND SECURITY

Jibin Joy[1], Dr. S. Devaraju[2]

[1] *Research Scholar (Ph.D.), Department of Computer Science, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India*

[2] *Assistant Professor, Department of Computer Applications, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India*

jibinjoysamuel@gmail.com[1], devamcet@gmail.com[2]

ABSTRACT:

Data deduplication is a crucial technique for packing data and reducing duplication when transferring data. It is widely used in the cloud to restrict the usage of capacity memory and aids in transmission capacity sparing. Before redistributing data, an encryption mechanism is used to ensure the integrity of sensitive data during the deduplication process. The SHA algorithm is being used to save data in text format. To generate the security bits, padding is appended to the text. In de-duplication, it calculates the hash, i.e. hexadecimal number, string, and integer data. Hash-based de-duplication is the implementation of whole file hashing to the entire file. Text data's hash values are considered to as feature properties. In contrast to traditional deduplication solutions, clients that transfer data to the cloud certify duplication inside the cloud data. In virtualization, both limiting primary memory size and memory blockage are considered important bottlenecks. Memory deduplication identifies pages with the same content and merges them into a single data file, reducing memory usage, memory parceling, and improving execution. In cloud storage, the MPT is used to deduplication so that it is used in single copies of the same data for different data owners. If any data users try to replicate the same data, it will be mapped and related to the archive data, implying that the data can't be stored away. To ensure cloud data security, encryption techniques are used to encrypt data throughout deduplication procedures and prior to outsourcing cloud data.

Keywords: Mapping Technique (MPT), Virtual Machine (VM), Content-Based Page Sharing (CBPS), in-line duplication check algorithm (HIDC)

I. INTRODUCTION

Cloud adaptively provides storage and preparation administrations, such as SaaS, IaaS, and PaaS, that powerfully expand the limit and add capacities without putting resources into new frameworks or allowing new programming. Distributed computing expands the current capabilities of Information Technology (IT) because the cloud adaptively provides storage and preparation administrations, such as SaaS, IaaS, and PaaS, that powerfully increase the limit and add capacities without putting resources into new frameworks or allowing new programming. Deduplication, which reduces storage costs by allowing us to keep only one identical copy of data with minor differences, has become increasingly vital as the global data reserve has grown dramatically. Before being re-appropriated, files are generally secured to ensure their secrecy. Traditional security will invariably result in multiple distinct cipher texts being generated from the same plaintext by various households. The mystery will take time, which will obstruct data deduplication. On the data target, the working assets can be effectively filtered out and about into a hypercube structure. The hypercube scales consistently across the board as assets are added or removed in response to changes in the number of supplied VM samples. Each process hub operates independently and deals with its remaining task by applying a lot of dispersed load adjusting criteria and calculations, without oversight from any focal segments. In a cloud data centre, servers are constantly over-provisioned in order to meet the highest demand for requests, waste a lot of energy in the process.

II. NEED FOR STUDY

In the sphere of technology, the world is on the verge of fast transformation. So, these technology applications are used in a variety of industries, including social networking, online money transactions, banking, and so on. As a result, communication between different systems is a vital aspect of technology, as different clients tend to share data or information. Main memory size limitation is considered as one of the major bottlenecks that cloud technology faces in virtualization environments. Cloud adaptively provides storage and preparation administrations, such as SaaS, IaaS, and PaaS, that powerfully expand the limit and add capacities without putting resources into new frameworks or allowing new programming. Distributed computing expands the current capabilities of Information Technology (IT) because the cloud adaptively provides storage and preparation administrations, such as SaaS, IaaS, and PaaS, that powerfully increase the limit and add capacities without putting resources into new frameworks or allowing new programming.

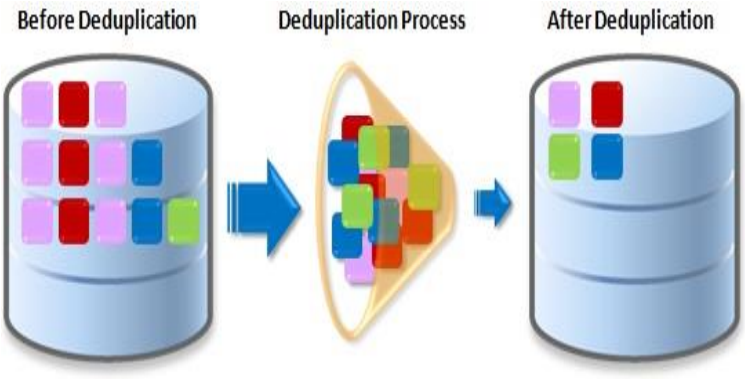

Figure 1.1

Due to the large amount of storage space available in the cloud, there is a problem with single users keeping a lot of redundant data within the cloud. A lot of cloud storage is wasted due to redundant data storage, and anytime a client tries to upload a file with a different name but identical data within, the file is successfully uploaded to the cloud storage. The biggest disadvantage of the current solution was redundant data in the cloud. The private/open cloud is used as an intermediate to allow clients or data owners to execute duplication confirmation safely and with different benefits. Such a strategy is practicable and has sparked a lot of discussion among experts. Each cloud client has their own cloud record, which is saved in their own storage space. For example, if a large number of cloud users have the same record to save, the cloud must store all of the documents. A lot of distributed storage is being squandered here. Because the rationale is that each cloud client's requirements for a similar document that they desire to store in the cloud are the same.

III. LITERATURE SURVEY

CMD: Classification-based Memory Deduplication through Page Access Characteristics by L. Chen, Z. Wei, Z. Cui, M. Chen, H. Pan, Y. Bao.

One of the most significant obstacles in virtualization situations is the limited principle memory size. CBPS (Content-Based Page Sharing) is a useful memory deduplication approach for reducing server memory requirements, in which pages with the identical content are identified and shared into a single duplicate. Kernel Same Page Merging (KSM) is a widely used CBPS technique that divides all memory pages into two global correlation trees (a steady tree and a precarious tree). To identify page sharing opportunities, every rival page should be compared to pages now in these two massive global trees. In any case, because the vast majority of pages have separate content, it will result in massive useless page correlations and, as a result, overwhelming overhead. The authors suggest CMD, a lightweight page Classification-based Memory Deduplication strategy, to reduce futile page correlation overhead and to differentiate page sharing open doors usefully in this study. The basic idea behind CMD is that sites are grouped into different groups based on how easy they are to access. Pages with similar access features are advised to have a higher possibility of having the same material and are thus grouped together. In CMD, the massive global correlation trees are divided into smaller trees with dedicated neighbourhood trees in each page order. Pages in CMD are divided into multiple groups based on page access attributes, and the massive global correlation trees are divided into several trees with dedicated ones in each page grouping. Page inspections are conducted in the same order, and pages from other orders are never compared or contrasted, as these are most likely spurious associations. To catch fine granularity page, get to attributes, the authors use a lightweight equipment-assisted memory follow checking framework.

Difference Engine: Harnessing Memory Redundancy in Virtual Machines by D. Gupta, S. Lee, M. Vrable, S. Savage, A. C. Snoeren, G. Varghese, G. M. Voelker, and A. Vahdat

For Internet facilitating centers and cloud-based process administrations, virtual machine screens are a well-known stage. Virtual machine screens reduce both the capital expenditure and the board overhead of facilitating focuses by multiplexing equipment assets across virtual machines (VMs) running item working frameworks. Fitting organization and movement methods can take advantage of factual multiplexing to make the most of available processors.

However, primary memory isn't amenable to such multiplexing and is frequently the sticking point to achieving higher levels of combination. Previous research has shown that content-based page sharing reduces the memory footprint of VMs running similar operating systems and apps unobtrusively. Primary memory space is one of the most important bottlenecks in higher examples of virtual device multiplexing. When operating homogenous extraordinary activities, the prior function shows which significant storage reserve memory is obtainable from collecting indistinguishable Web Pages across more than virtual machines. The reason for this study is because locating and repairing comparable pages and in-memory page pressure can provide significant additional memory reserve funds. The authors provide the design and evaluation of Difference Engine in order to demonstrate the possible memory savings that can be obtained by combining complete page sharing, page fixing, and pressure.

The authors examine our experience tending to various specialized difficulties, including:

i) calculations to quickly identify candidate pages for patching,

ii) request paging to support over-subscription of the total assigned physical memory, and

iii) a clock mechanism to identify exact target machine pages for patching, sharing, compression, and paging.

Data Deduplication with Encrypted Big Data Management in Cloud Computing, Nahlah Aslam K.P., Dr. Swaraj K.P.

This author suggests a cloud-based deduplication solution that can also handle large amounts of data. This study attempts to explain cloud resistance to brute force attacks, as well as how it manages storage through deduplication without the direct involvement of data owners (user). The proposed system is primarily based on paragraph-wise token generation of text data, symmetric encryption and re-encryption to manage encrypted data storage with deduplication in the cloud, and symmetric encryption and re-encryption to manage encrypted data storage with deduplication in the cloud. To avoid data duplication, the author employs symmetric encryption. A symmetric key is generated using any standard key generation procedure, and the message to be encrypted is given as input in the Symmetric encryption technique. This algorithm creates a cypher message by encrypting that specific message with the specified key.

This procedure is performed by the user to protect their message from unauthorised access. The same generated key that was used for encryption can be used to decrypt this encrypted message. The user uses this decryption method to decrypt encrypted data stored in cloud storage. Deduplication is the most widely used method for reducing storage costs, bandwidth, and energy consumption. Many existing systems are geared toward file-level deduplication and do not provide deduplication of encrypted data. As a result, the author aspires for file content block-level cloud deduplication. The proposed method seeks to find the percentage of identical files that are duplicated in the same system. The system uses a deduplication technique on files that have an 80% resemblance to currently existing files. The suggested system's security is ensured by the use of symmetric key encryption and re-encryption theory.

An Effective Data Storage Model for Cloud Databases using Temporal Data De-duplication, S. Muthurajkumar, M. Vijayalakshmi, A. Kannan

This work focuses on leveraging Temporal Data De-duplication to develop a Data Storage Model for Cloud Databases. The author uses a bespoke architecture that consists of nine components: User and clients, Node Server, Temporal Information Manager, Rule Base, File System Control Manager, Data Storage Node, Access Control Monitor, Role Database, and Cloud Data Center. For greater speed, a new method called Temporal Data Deduplication Algorithm (TDDdA) was implemented. This strategy focuses on allocating and supplying user responsibilities on a regular basis, while enabling temporal limits to improve system security indirectly. The Eucalyptus cloud utility was used to do the performance analysis. To provide security, the Java platform employs an Elgama-based public key algorithm. An effective data storage system has been proposed for higher space efficiency by the author. Furthermore, this work has been integrated into the present real-time scenario for auditing files and allowing the user to authenticate the content of those files. The obfuscation method, which can be implemented in instruct to improve performance efficiency, is another option to provide security to this system. Integrating obfuscation and encryption techniques will improve the level of confidentiality. It aids in the protection of data in the storage cloud from attacks from both internal and external clients.

Enhanced Storage Optimization System (SoS) for IaaS Cloud StorageS. Muthurajkumar, M. Augustus Devarajan A, SudalaiMuthu T

In this research, the customized Storage Optimization System (SoS) employs deduplication as a preprocessing strategy for file/data storage in IaaS cloud storage. The deduplication process primarily employs four steps. Chunking comes first, followed by Deduplication Checking, Metadata Management, and Data Storage. Chunking: The entire clustering element is broken down into little chunks or package units to analyze the data element more quickly and efficiently. This procedure aids in the detection of repeated data elements associated with various files or data patterns, which are then collected and arranged as structure-based metadata. The non-overlapping blocks are chunked using the SHA-1 or

MD5 methods, and these blocks are divided into fixed or variable size chunking packages. Deduplication Checking: Using a chunk comparison, similar data can be deleted. The index reference table will be compared by the hash key in the hash key algorithm value to determine the occurrence of duplicate files. If data repetition is discovered, the data units are not saved to the hard drive (storage) and are erased. If the hash key value does not match, the data files will be archived and synchronized according to the hash key values Metadata Lookup Management: The hash key values on each block of data segments on file with numerous indexing parameters are included in the metadata schema on the SHA-1 algorithm mapping operation. For each stage comparison, the Path Code Hash ("DPCH") with imbalanced metadata distribution and access point token will be captured, as well as the original file is rebuilt. Data Storage: In order to temporarily store data, RAM chips are arranged using a metadata structure with a hash value index. The metadata structure is updated through APIs by improving the hash index reference values in the containers. The file segmentation process was constructed using the metadata structure and hash values. The data structure linked list idea was used to recalculate all of the parameterized hash values. During the configuration process, the file data elements' bin structure is altered, resulting in distinct chunk segments with a certain range of connections within file segmentation. The metadata for the saved files has been determined and is being updated in real-time.

An Efficient Data Duplication System based on Hadoop Distributed File System, D.Veeraiah, J.Nageswara Rao

The Duplication Management System in this paper stores duplication files in HDFS. The data files are divided into two categories by the system: cold and hot. After the data is recognized, the duplication ratio for Hot Data improves. To ensure quality, erasure coding is extended to Cold Data and such actions are handled through the HDFS Login Process. For the duplication methodology, the overall method splits each data into two groups, hot data, and cold data, based on their priority in the Hadoop cluster. Different characteristics are measured to determine the value of a file, such as the number of users accessing the file, the number of nodes accessing the file, and the file's redundancy rate. In HDFS, user log files are analyzed to determine a value. The user log files in HDFS are evaluated to determine certain values. The popularity of each separate file is calculated after evaluating and retrieving the relevant values.

Cloud based Hybrid Model for Authorized Deduplication Miss Prachi D. Thakar, Dr. D.G.Harkut

By integrating differential authorization to users in the duplicate check, the authors suggest a model that can protect the data. The owner of the file will determine what privileges should be granted to another user and who will have access to that file. The hybrid cloud idea is used to protect sensitive data in this paradigm. One private and one public cloud are employed in the hybrid cloud idea. First, the client must engage with the private cloud session. Only the private cloud is used for duplicate data verification and key distribution. Because a generated key is utilized in this technique, privacy is preserved. The public cloud is only used for storing data. The file key is necessary for encryption, and the key is generated on a private cloud. Simple file-level deduplication is offered for space utilization when the file is kept in encrypted form in a public cloud service. The AES algorithm is used to reduce encryption and decryption calculation time.

Balancing DRAM Locality and Parallelism in Shared Memory CMP Systems, Min Kyu Jeong, Doe Hyun Yoony, Dam Sunwooz, Michael Sullivan, Ikhwan Lee, and Mattan Erez

The authors look at innovative strategies for reducing obstacles in between string accesses. The authors propose partitioning the inner memory banks between centers to decouple their entrance streams and eliminate territory impedance. The authors accomplish this by expanding the OS's physical edge allotment calculation such that physical casings mapped to the same DRAM bank can only be distributed to a single string. The method balances the conflicting demands of area and parallelism. It relies on bank apportioning, which assigns a large number of autonomous banks to each potentially clashing string using OS-controlled coloring. To compensate for the lower bank parallelism available to each string, the authors use DRAM sub-positioning to increase the number of banks in the framework without increasing the cost. This centralized bank division and sub-positioning technique enable us to primarily support execution and productivity while maintaining reasonableness.

This technique's ability to improve territory is demonstrated through assessment, which leads to improved execution and proficiency. Furthermore, when evaluating framework configurations with few banks for each string, the paper shows that the technique is substantially more progressive. Expect future frameworks to use such bank-obliged designs, given the often prohibitive cost of expanding memory banks compared to the cost of increasing the number of simultaneous strings in the CPU. When all applications in a multi-modified remaining load have a large spatial region, bank parceling works well.

Cloud iDedup: History aware In-line Deduplication for Cloud storage to Reduce Fragmentation by Utilizing Cache Knowledge Reshma A. Fegade, R.D.Bharati,

For deduplication and erasure coding purposes, the author suggests the in-line duplication check method (HIDC). Erasure coding aids in the encoding of data, which is subsequently compressed and saved on cloud servers. In this example, one server serves as a backup, storing encoded files that users can decode and download in the event of a primary server failure or another loss event. During the data backup process, the Sparse Container is utilised to store the data blocks using the in-line duplication check technique (HIDC). When data hashes are compared against historical hash values for backup purposes, the data hashes are matched. The data is discarded from the container if the hash values are successfully matched. A container is a dynamic folder that exists on cloud servers. It serves as a data buffer throughout the process. Two servers were utilized in the implementation, one without erasure and the other with erasure coding. Server one saves files without encoding them, therefore it takes up more space than server two. Server two reduces the amount of data and blank spaces in the data files.

Improving System Throughput and Fairness Simultaneously in Shared Memory CMP Systems via Dynamic Bank Partitioning Mingli Xie, Dong Tong, Kan Huang, Xu Cheng

Memory channel division divides the data of strings that are likely to collide with one another into different channels. However, it unjustly distributes memory assets and physically fuels memory contention of escalating strings, resulting in an extended log jam of these strings and significant framework injustice. Bank parceling divides memory banks between centers and eliminates blockage.

However, historical similar bank apportioning restricts the number of banks available to each string and reduces bank-level parallelism. Initially, the authors suggest Dynamic Bank Partitioning (DBP), which allocates memory banks based on string requirements for bank amounts.

DBP compensates for the loss of bank-level parallelism caused by equivalent bank division. The main rule is to profile strings' memory characteristics at run-time and gauge their bank sum requests, then use the estimation to coordinate our bank splitting. Bank parceling divides memory banks between centers and eliminates string impedance. Past bank division allows an equal number of memory banks to each center without considering the differences in bank-level parallelism needs for particular strings. It occasionally limits the number of banks available to an application and reduces bank-level parallelism, essentially distorting framework execution. The authors propose Dynamic Bank Partitioning, which evaluates strings' requests for bank-level parallelism to compensate for the reduced bank-level parallelism. DBP increases framework throughput while ensuring that it is fair. Memory access planning focuses on optimizing framework performance by taking into account string memory access and framework decency. Saving column support territory is the focus of bank division, which solves the between-string impedance problem. These two strategies are symmetrical and reciprocal to one another. The improved spatial area can benefit memory booking, and better planning can benefit bank parceling. In this way, the authors present an extensive methodology that incorporates Dynamic Bank Partitioning and Thread Cluster Memory booking.

Memory Latency Reduction via Thread Throttling by H. Cheng, C. Lin, J. Li, and C. Yang

To boost framework execution, the authors create a memory string throttling instrument that tunes the appropriate memory strings powerfully under high burden variety. For stage discovery, the suggested run-time instrument examines a program's memory and calculation proportions. It then chooses the memory string required for the next project stage based on an explanatory model that can evaluate framework execution under various constraint values.

To show the concept, the authors model the component in certain real-world applications, just as if it were still being made. The writers evaluate their exhibits using real machines. Furthermore, careful run-time planning results in a geometric mean of 12% improvement in execution time for real-world programs on equivalent hardware. The authors construct a logical model to capture the substance of on-chip figure and off-chip correspondence to better comprehend the presentation speedup at varied memory assignment booking criteria. The authors certify the scientific model with exploratory outcomes from genuine equipment.

Based on this logical model, the authors devise a scheme to detect stage changes in applications and select an appropriate booking technique for each stage to address the memory divider problem. With a pool of fabricated remaining burdens, our results appear to be 20 percent faster. On the Intel i7 quad-center computer, our runtime instrument showed a 12 percent speedup for realistic benchmarks. Affectability tests of the suggested system with shifted checking the time, input working sets, program stages, and equipment settings reveal that the proposed run-time component is effective and performs similarly to a detached thorough hunt instrument. In addition, the proposed component gives 5% higher speed up by and large than a gullible strategy dependent on online comprehensive hunt. The higher speedup of the proposed system originates from less run-time observing overhead and better determination of the booking strategy for each stage.

Utility-Based Cache Partitioning: A Low-Overhead, High-Performance, Runtime Mechanism to Partition Shared Caches Moinuddin K. Qureshi Yale N. Patt

The growing gap between microprocessor speed and DRAM speed is a serious problem for PC designers. It's critical to boost DRAM's speed and throughput to close the gap. Furthermore, on multi-center stages, the DRAM memory shared by all centres is prone to memory contention and obstruction, which might result in actual execution debasement among parallel running strings. To solve these challenges, this study suggests systems that combine the two advantageous situations of parcelling centres, strings, and memory banks into gathers to reduce impedance across various gatherings and to group memory gets to of a same line together to reduce reserve miss rate. This research proposes a memory advancement structure called consolidated string booking with memory planning (CTMS), which limits memory access plan length, memory access time, and reduces obstruction to improve multi-center framework execution. CTMS achieves this by dividing centers, strings, and memory banks into three groups: center gathering, string gathering, and bank bunch. By banding string gathering to center gathering and bank mindful page distribution, the three groups of center, string, and bank gathering are reliance shaped into four free subsystems, reducing the obstruction between strings running different center gatherings. Join the BFS computation for even better performance by limiting the length of memory access plans and lowering memory access times. CTMS limits memory access plan length improves memory access time, and reduces impedance for enhancing the execution of multi-center implanted frameworks, according to preliminary findings.

Singleton: System-wide Page Deduplication in Virtual Environments Prateek Sharma Purushottam Kulkarni

The issue of memory management in hypervisors is addressed, and a KVM(Kernel Virtual Machine)-based system-wide page deduplication solution is proposed to improve memory utilization efficiency. We specifically address the issue of double caching in KVM, where the same disc blocks are cached in both the host (hypervisor) and guest (VM) page caches. The major features of Singleton are near-identical page sharing across guest virtual machines and the deployment of an exclusive cache for the host and guest page cache hierarchy. To identify and share pages across guest virtual machines, we use and will try to improve KSM (Kernel Same Page Merging). We use guest memory snapshots to clean the host page cache and keep a single copy of a page across the host and guests. We do not modify the guest or make any assumptions about its behavior because Singleton runs on a fully black-box premise. The authors demonstrate how traditional operating system cache management strategies are inadequate for virtual settings, as well as how Singleton supplements and improves the existing Linux kernel memory management mechanisms.

By lowering the size of the host cache (up to an order of magnitude) and raising the cache-hit ratio, Singleton can optimize its utilization (by a factor of 2x). This corresponds to speedier I/O for the virtual machine (by 40%). Singleton's unified page deduplication and host cache scrubbing method can free up a lot of RAM and allow for more memory over commitment. The optimizations to page deduplication, author implemented keep the overhead down to less than 20% approx. CPU utilization.

Enhancing Operating System Support for Multicore Processors by Using Hardware Performance Monitoring Reza Azimi David K. Tam Livio Soares Michael Stumm

The authors present three contextual analyses on how a multicore-aware working framework can use these online capacities for determining reserve segment sizes, which reduces conflict in the common store among applications, identifying memory districts with poor reserve utilisation, which aids in isolating these areas to reduce store contamination, and identifying string sharing. Multicore processors handle key issues that have arisen during the evolution of processor models, such as the memory divider, control divider, and multifarious nature of the structure.

This fundamental shift in processor engineering necessitates extensive framework programming assistance to achieve optimum capacity in terms of execution and power efficiency.

PMUs provide fine-grained and consistent information about a running framework's presentation. The authors have demonstrated how simple features like equipment information examining may be used to depict reserve conflict and contamination in on-chip shared storage, as well as to identify sharing examples among strings.

Because of the features available in the IBM POWER5 multi-core CPU, such as information inspection and slowdown breakdown, the designers have implemented three runtime enhancement methodologies in the working framework, all of which result in significant performance improvements.

Managing Performance Overhead of Virtual Machines in Cloud Computing: A Survey, State of the Art,and Future Directions Fei Xu, Fangming Liu,

The live migration of virtual machines serves as the important factor of managing the computing resources and problems in the performance overhead of virtual machines in IaaS clouds. Live virtual machine migration is successful in controlling virtual machine performance overhead in many circumstances, ranging from single server virtualization to a single data center, in addition to activating normal processes like load balancing and power saving.

In the IaaS cloud network resource and shared storage resource, managing the performance overhead of virtual machines induced by live virtual machine migration is critical to ensuring overall virtual machine performance. Existing solutions largely rely on two elements to improve the speed of live Virtual Machine migration in a LAN or WAN: reducing the quantity of sent data during live virtual machine migration and lowering the memory or disc unclean rate.

Enhanced Cloud Data Security Using AES Algorithm Akhil K.M, Praveen Kumar M, Pushpa B.R

The author of this research attempts to present a fresh cloud data security model to address many inefficiencies in the current cloud security mechanism. It focuses on providing security to data transport through encryption techniques and makes the data un-accessible to other parties, in addition to data storage security. Advanced Encryption Standard (AES) is used to ensure data transfer security. It can encrypt a large amount of data and takes only a short amount of time to execute. Because users may send huge files, an encryption technology that is compatible with all data is used. Encryption and decryption are done at the user's end before being sent to the cloud server for storage and retrieval. The problem that has arisen as a result of granting third-party auditors data access is addressed by limiting the third-data party's access and limiting its role in verifying the validity of users during data requests. Because both encryption and decryption take place on the user's end, the chance of a third party accessing the data is ruled out.

A Framework Based on RSA and AES Encryption Algorithms for Cloud Computing Services Nasrin Khanezaei, Zurina Mohd Hanapi

The subject of data security in cloud storage was explored in this study, and cryptography techniques were employed to assure the security of data transmission and storage between users and cloud services. The paper focuses on the delivery of secure files between these two entities. This methodology used a combination of asymmetric and symmetric encryption techniques (i.e., RSA and AES encryption methods) to provide cloud data security. The emphasis was on employing the RSA encryption approach to make it difficult for attackers, as well as using AES encryption to reduce transmission time. Both encryptions were used in the process of sending files to the cloud and recovering files from the cloud. Because of the key distribution issue, symmetric encryption was utilized to retrieve cloud data. On the other side, it yields the best results. If the file size is greater than 254 bytes, the encryption process is doubled. Furthermore, repeating both processes for each file will result in system overhead. A hybrid encryption method like this is more secure than the existing method.

Enhanced RSA Algorithm with varying Key Sizes for Data Security in Cloud Dr. D.I. George Amalarethinam, H. M. Leena

For the next generation of large-scale systems, such as the cloud, security and performance are critical. The author proposes Division and Replication of Data in the Cloud for Optimal Performance and Security (DROPS), which splits user files into small ports and replicates them at needed locations inside the cloud to address the issue of security and performance as a secure data replication challenge. Splitting of a file into small fragments is performed based on given user criteria such that each fragment does not contain any meaningful data information. Each cloud node contains a different fragment to increase the data security in the cloud. An attack on a single node must not reveal the locations of other cloud fragments. To keep uncertainty about the locations of the file fragments for the attacker and to further improve the security, selects the nodes in a manner that they are not adjacent and are maintaining a certain distance from each other. The node distance is ensured by the means of the T-coloring. The nodes are chosen based on centrality measures that sustain and ensure an enhanced access time by employing 10 heuristics-based replication strategies as comparison techniques to the DROPS (Division and Replication of Data in Cloud for Optimal Performance and Security) methodology.

Applying Encryption Algorithm for Data Security in Cloud Storage Zaid Kartit, Ali Azougaghe, H. Kamal Idrissi, M. El Marraki, M. Hedabou, M. Belkasmi and A. Kartit

The author's proposed technique advises that files uploaded to the cloud be encrypted. The integrity and secrecy of the user-uploaded data are double-encrypted, and access to the data is granted only after each successful authentication. The AES algorithm will encrypt any existing files on the device. To improve security, the AES key will be encrypted with the RSA technique and stored on an internal server. Only the authorized user has access to download and view the encrypted files that have been uploaded to the system. The author of this work proposes a solution for data storage in an open cloud. Implementing this algorithm ensures data security. The data in the cloud can only be accessed by authorized users. Even if an attacker obtains the data by accident or on purpose, the intruder will be unable to decrypt it without two keys from two different locations. The AES encryption key is used for the encryption process of the data and is encrypted by the RSA-1024 algorithm robust and has never been broken. The data decryption requires double authentication and authorized users must have access rights to the company's server as well as cloud storage.

IV. SUMMARY

Data Storage Model for Cloud Databases using Temporal Data De-duplication [4] helps to drop the memory utilization up to 5% than the normal cloud usage but only 10% security is attained in this system during the process. Enhanced Storage Optimization System [5] improves file transfer performance.11.91% reduction in storage size usage than the other existing system, in turn, the bandwidth utilization also reduced up to 4.28% but, works only on structured file structure. It cannot handle unstructured file structures. An efficient Data Duplication System based on Hadoop Distributed File System [6] increases storage capacity utilization by up to 45% and storage space is optimized by implementing the cold data duplication factor and adding erasure code. Cloud-based Hybrid Model for Authorized Deduplication [7] provides secret keys for cloud server storage using the AES algorithm but can be implemented only in text files. Balancing DRAM Locality and Parallelism in Shared Memory CMP Systems [8] segment the inner memory banks between centers to disconnect their entrance streams and take out territory obstruction. balances the conflicting needs of locality and parallelism. But system configurations inherently have a small number of banks per thread. High memory bank configuration leads to high cost in implementation. Cloud iDedup [9] erasure coding for recovery and inline deduplication for cloud backup storage. The system utilizes less space thus minimize cost and recovery from data loss. Since it can be implemented only on text files, efficient implementation on cloud storage will not make much satisfactory. Dynamic Bank Partitioning [10] divides memory banks among cores and eliminates inter-thread interference. The overall performance improvement of the system is slight and sometimes it even affects system performance. Memory Latency Reduction via Thread Throttling [11] improves system performance on multi-core architecture by memory thread throttling mechanism. Performance is a little bit low in high-end processor architecture. Utility-Based Cache Partitioning [12] divides the cache among competing applications so that system increases Cache Partitioning in dual-core systems up to 23% for multi-programmed workloads. System-wide Page Deduplication in Virtual Environments [13] increases the cache-hit ratio double than the normal cache hit ratio and decreases CPU overhead by about 20%. Not possible for a brute-force scanner to detect identical pages efficiently. By Using Hardware Performance Monitoring [14] fully dependent on hardware performance which improves IPC (instructions-per-cycle) due to cache partition sizing. RSA and AES Encryption Algorithms for Cloud Computing Services [17] to achieve cloud data security focus on RSA encryption technique to provide difficulty for intruders as well as reducing transmission time by using AES encryption. The process of transferring files to the cloud and retrieving files from the cloud was accomplished by both encryptions respectively. Encryption and decryption [18] time is enhanced by dividing the file into several blocks by the ERSA Encryption algorithm but computation is complex. Division and Replication of Data in Cloud for Optimal Performance and Security [19] collectively deal with the security and performance in terms of retrieval time and nodes were separated employing T-coloring Automatic update mechanism that can identify and update the required fragments are not available. Applying Encryption Algorithm for Data Security in Cloud Storage [20] provides double security for data in the open cloud. Data upload and download from the open cloud using this algorithm is a little time-consuming.

Table 1.1

PAPER NAME	**METHODOLOGY**	**MERITS**	**DEMERITS**
Balancing DRAM Locality and Parallelism in Shared Memory CMP Systems by M. Jeong, D. Yoon, D. Sunwoo, M. Sullivan, I. Lee, and M. Erez. [8]	propose to segment the inner memory banks between centers disconnect their entrance streams and take out territory obstruction	Balances the conflicting needs of locality and parallelism	Area is obliged by the constrained planning support size
Improving System Throughput and Fairness Simultaneously in Shared Memory CMP Systems Via Dynamic Bank Partitioning by M. Xie, D. Tong, K. Huang and X. Cheng. [10]	Propose a Dynamic Bank Partitioning (DBP), which allotments memory banks as per strings' prerequisites for bank sums	Divide's memory banks among cores and eliminates inter-thread interference	The adequacy in recouping region of memory planning is confined because of the constrained booking cradle size
Utility-based cache partitioning: A low-overhead, high-performance, runtime mechanism to partition shared caches by M. K. Qureshi, and Y. N. Patt[12]	Implements utility-based cache partitioning	Utility-Based Cache Partitioning (UCP) to divide the cache among competing applications so that system increases Cache Partitioning in dual-core systems up to 23%.	An application gets from store assets may not straightforwardly associate with its interest for the reserve asset
An Effective Data Storage Model for Cloud Databases using Temporal Data De-duplication [4]	Cloud Databases using Temporal Data De-duplication	It helps to drop the memory utilization up to 5% than the normal cloud usage	Only 10% security is attained in this proposed system during the process.

Enhanced Storage Optimization System (SoS) for IaaS Cloud Storage[5]	deduplication technique as a preprocessing of file/data storage in the IaaS storage	The proposed SoS system improved in file transfer performance.11.91% reduction in storage size usage than the other existing system, in turn, the bandwidth utilization also reduced up to 4.28%.	The proposed SoS system works only on structured file structure. It cannot handle unstructured file structures.
An Efficient Data Duplication System based on Hadoop Distributed File System [6]	The method divides each data into two groups based on their priority in the Hadoop cluster, as hot data or cold data	The proposed approach increases storage capacity utilization of up to 45%. Storage space is optimized by implementing the cold data duplication factor and adding erasure code.	The proposed approach doesn’t give importance to the file content. Replication position strategy in the Hadoop cluster is much better than the Hadoop space utilization approach
Singleton: System-wide Page Deduplication in Virtual Environments [13]	KVM(Kernel Virtual Machine))-based system-wide page deduplication solution to increase memory usage efficiency	Increasing the cache-hit ratio double than the normal cache hit ratio and decreases CPU overhead by about 20%	Not possible for a brute-force scanner to detect identical pages efficiently

V. CONCLUSION

In today's essential and optional stockpiling frameworks, data deduplication is commonly applied to boost capacity proficiency. Each of the papers mentioned above will discuss the benefits and drawbacks of each based on functionality, security, and data storage. One of the biggest issues that current technology faces in day-to-day operations are security. In a cloud environment, data de-duplication is one of the techniques for avoiding duplicate data in memory or the cloud. This strategy, as indicated in the previous articles, is widely utilized in various fields to avoid data duplication. Similarly, the loss of a single such basic information chunk may theoretically render all reference (sharing) records unreachable. The mapping approach is the same for all clients who attempt to access comparable records and have varying levels of access. For future study, several encryption techniques can be examined efficiently to overcome the duplication of a significant amount of data by different users within the cloud environment using MPT (memory mapping technique) and to ensure the security of cloud data.

VI. REFERENCES

1. CMD: Classification-based Memory Deduplication through Page Access Characteristics by L. Chen, Z. Wei, Z. Cui, M. Chen, H. Pan, Y. Bao.
2. Difference Engine: Harnessing Memory Redundancy in Virtual Machines by D. Gupta, S. Lee, M. Vrable, S. Savage, A. C. Snoeren, G. Varghese, G. M. Voelker, and A. Vahdat
3. Data Deduplication with Encrypted Big Data Management in Cloud Computing, Nahlah Aslam K.P., Dr. Swaraj K.P. (2019- IEEE Xplore ISBN: 978-1-7281-1261-9)
4. An Effective Data Storage Model for Cloud Databases using Temporal Data De-duplication, S. Muthurajkumar, M. Vijayalakshmi, A. Kannan (2016 IEEE Eighth International Conference on Advanced Computing (ICoAC), 978-1-5090-5888-4/16/$31.00@2016 IEEE)

5. Enhanced Storage Optimization System (SoS) for IaaS Cloud StorageS. Muthurajkumar, M. Augustus Devarajan A, SudalaiMuthu T (2020, Proceedings of the Fourth International Conference on Inventive Systems and Control (ICISC 2020) .IEEE Xplore Part Number: CFP20J06-ART; ISBN: 978-1-7281-2813-9)
6. An Efficient Data Duplication System based on Hadoop Distributed File System, D.Veeraiah, J.Nageswara Rao(2020, Proceedings of the Fifth International Conference on Inventive Computation Technologies (ICICT-2020) IEEE Xplore Part Number:CFP20F70-ART; ISBN:978-1-7281-4685-0)
7. Cloud based Hybrid Model for Authorized Deduplication Miss Prachi D. Thakar, Dr. D.G.Harkut
8. Balancing DRAM Locality and Parallelism in Shared Memory CMP Systems, Min Kyu Jeong, Doe Hyun Yoony, Dam Sunwooz, Michael Sullivan, Ikhwan Lee, and Mattan Erez
9. Cloud iDedup: History aware In-line Deduplication for Cloud storage to Reduce Fragmentation by Utilizing Cache Knowledge (2016, Reshma A. Fegade, R.D.Bharati, 978-1-5090-1338-8/16/$31.00 ©2016 IEEE)
10. Improving System Throughput and Fairness Simultaneously in Shared Memory CMP Systems via Dynamic Bank Partitioning, Mingli Xie, Dong Tong, Kan Huang, Xu Cheng 978-1-4799-3097-5 ©2014 IEEE
11. Memory Latency Reduction via Thread Throttling by H. Cheng, C. Lin, J. Li, and C. Yang 2010, IEEE, DOI 10.1109/MICRO.2010.39)
12. Utility-Based Cache Partitioning: A Low-Overhead, High-Performance, Runtime Mechanism to Partition Shared Caches Moinuddin K. Qureshi Yale N. Patt(2006, IEEE, 10.1109/MICRO.2006.49)
13. Singleton: System-wide Page Deduplication in Virtual Environments Prateek Sharma Purushottam Kulkarni (2012, ResearchGate, 10.1145/2287076.2287081)
14. Enhancing Operating System Support for Multicore Processors by Using Hardware Performance Monitoring (2009, ResearchGate, DOI:10.1145/1531793.1531803)
15. Managing Performance Overhead of Virtual Machines in Cloud Computing: A Survey, State of the Art,and Future Directions (2014, IEEE, DOI: 10.1109/JPROC.2013.2287711)
16. Enhanced Cloud Data Security Using AES Algorithm (2014, IEEE, DOI: 10.1109/JPROC.2013.2287711)
17. A Framework Based on RSA and AES Encryption Algorithms for Cloud Computing Services (2014, IEEE, DOI: 10.1109/I2C2.2017.8321820)
18. Enhanced RSA Algorithm with varying Key Sizes for Data Security in Cloud (2017, IEEE, DOI: 10.1109/WCCCT.2016.50)
19. DROPS: Division and Replication of Data in Cloud for Optimal Performance and Security (2015, IEEE, DOI: 10.1109/TCC.2015.2400460)
20. Applying Encryption Algorithm for Data Security in Cloud Storage (2016, ResearchGate, Springer, DOI: 10.1007/978-981-287-990-5_12)