Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 4 (2023)

Ensuring Data Integrity And Security In Diverse Cloud Environments To Prevent Duplicacy.

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

[1] Research Scholar (Ph.D.), Department of Computer Science, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India
[2] Senior Assistant Professor, VIT Bhopal University, Bhopal, Madhya Pradesh India
E-mail: [1]jibinjoysamuel@gmail.com, [2]devamcet@gmail.com

Abstract: Data deduplication is a valuable technique for compressing and minimizing data duplication during data transfers, especially in cloud environments. By eliminating redundant data, it optimizes transmission capacity and reduces memory usage. To ensure the integrity of sensitive data, encryption is applied throughout the deduplication process. The SHA algorithm is commonly used for storing text data during deduplication. It generates security bits by padding the text and computes a hash consisting of hexadecimal, string, and integer data. Hash-based deduplication involves hashing the entire file and treating the hash values of text data as unique identifiers. This allows clients to identify duplicate data within the cloud. The Proof-of-Work (PoW) algorithm is widely utilized in blockchain networks like Bitcoin and Ethereum. Its primary function is to verify transactions through a process called mining. Miners engage in competition to solve complex mathematical problems, and the first one to find a solution is granted the right to add a new block to the blockchain. PoW relies on cryptographic hashing, such as the SHA-256 hashing function, to validate and secure transactions. On the other hand, the Proof of Retrievability (PoR) algorithm finds application in cloud computing systems. It serves as a consensus mechanism, ensuring that cloud storage providers store and retrieve data accurately. PoR enables cloud providers to demonstrate to consumers that their files can be fully recovered. This algorithm incorporates cryptographic proofs that validate the integrity and availability of the stored data. In cloud storage, deduplication is often implemented using the Memory mapping technique (MPT). This allows multiple data owners to store the same data in a single copy, enhancing storage efficiency. To maintain data security, encryption is applied both before and during the deduplication process, ensuring that sensitive information remains protected. In summary, data deduplication is a powerful method for compressing data, minimizing duplication, and optimizing transmission capacity. With encryption techniques and hash-based identification, it provides secure and efficient deduplication in cloud environments, while memory deduplication and MPT support further enhance performance and storage efficiency.

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

1. Introduction

SaaS, IaaS, and PaaS, which are cloud storage and preparation administrations, adaptively increase the cap and enhance capabilities without requiring new frameworks or allowing new programming. The cloud adaptably offers storage and preparation administrations like SaaS, IaaS, and PaaS that greatly increase the limit and extend capabilities without spending money on new frameworks or enabling new programming. Distributed computing expands the current capabilities of Information Technology (IT) because the cloud adaptively provides storage and preparation administrations like SaaS, IaaS, and PaaS that significantly enhance the limit and add capacities without spending resources in new frameworks. Deduplication, which lowers storage costs by enabling us to maintain only one identical duplicate of data with minute changes, has grown in significance as the world's data reservoir has increased significantly. Before being reappropriated, files are typically protected to protect their privacy. Traditional security will always lead to distinct cypher messages being produced from the same plaintext by various families. The riddle will take time, and it will hinder data deduplication. On the data target, working assets can be successfully filtered out and put into a hypercube structure. The hypercube scales uniformly as assets are added or removed in response to changes in the quantity of a given VM sample. Each process hub is self-

contained and handles its remaining tasks using various distributed load adjustment criteria and algorithms, with no oversight from focus segments. In a cloud data centre, servers are always over-provisioned to satisfy the highest demand for requests, which wastes a lot of energy.

2. Review

In a research study [1], it was found that limited primary memory size poses a significant challenge in virtualization settings. To address this issue, two memory deduplication techniques called Content-Based Page Sharing (CBPS) and Kernel Same Page Merging (KSM) were investigated. CBPS works by detecting and consolidating pages with identical content into a single duplicate, thereby reducing the memory requirements of the server. On the other hand, KSM maintains two global correlation trees, namely a stable tree and a volatile tree, to preserve all memory pages. While CBPS and KSM are effective in memory deduplication, there are challenges associated with identifying page-sharing opportunities. This is because a large number of pages contain unique information, resulting in significant overhead from unnecessary page correlations.

In order to address the challenge of unnecessary page correlation overhead, the study introduces a lightweight approach called Classification based Memory Deduplication (CMD). This method focuses on improving the efficiency of page classification to avoid unnecessary comparisons. Unlike the traditional approach of exhaustively comparing each engaging page with pages in the global correlation trees, CMD adopts a classification-based strategy to swiftly identify potential opportunities for page sharing. By leveraging this classification-based approach, CMD aims to significantly reduce the overhead associated with page correlations. Rather than performing time-consuming comparisons for every page, CMD efficiently categorizes pages based on their characteristics and similarities. This enables the identification of potential page-sharing opportunities in a more streamlined and efficient manner. The introduction of CMD provides a more effective and lightweight solution for memory deduplication, particularly in virtualization settings with limited primary memory. This approach enhances the overall efficiency of the deduplication process, helping to optimize memory usage and improve the performance of virtualized environments.

The implementation of CMD aims to optimize memory deduplication by reducing the overhead associated with unnecessary page correlations. This approach provides a more efficient and lightweight solution for memory deduplication in virtualization settings. CMD categorizes pages based on their accessibility, assuming that pages with similar access patterns are likely to contain the same information. By grouping pages together and creating dedicated local trees for each page category, the large global correlation trees in CMD are divided into numerous smaller trees. This allows for efficient page comparisons and eliminates the need to compare pages from unrelated categories, reducing unnecessary overhead. The authors of the study employed a memory trace-based methodology to capture fine-grained page access attributes. By using basic tools, they were able to gather information about page access patterns and create different trees based on these patterns. This approach improves the efficiency of page examinations and enhances the overall performance of the memory deduplication process. In another research paper [2], it is mentioned that virtual machine monitors (VMMs) are widely used in cloud-based process management and hosting services. VMMs help lower the capital expenditure and management overhead of data centers by multiplexing hardware resources among virtual machines running different operating systems. However, one of the main challenges in achieving higher levels of consolidation is the limited primary memory. Previous research has investigated content-based page sharing as a means to reduce the memory usage of virtual machines that run similar operating systems and applications. This approach involves identifying pages with identical content and sharing them among multiple virtual machines. By implementing techniques such as in-memory page pressure and sub-page level sharing through page pinning, notable enhancements in performance can be realized. This emphasizes the significance of identifying and resolving similar pages and efficiently managing memory resources.

To demonstrate the potential memory savings, the authors present the design and evaluation of the Difference Engine. This system combines techniques such as page sharing, page pinning, and pressure to optimize memory utilization in virtual device multiplexing scenarios. By implementing these strategies, substantial memory savings can be achieved, further enhancing the efficiency of virtualization environments. The authors look at our experience dealing with a variety of specialised issues, such as

1. calculations to quickly identify incoming pages for fixing.

2. request paging to help with over-membership of absolute designated physical memory, and
3. a clock system to recognise appropriate objective machine pages for sharing, fixing, pressure, and paging.

According to Research Paper [40], current memory architectures rely on the spatial domain to achieve high data transfer capacity while managing power and cost constraints. However, as the number of memory-providing centers increases, the access streams from different threads become intertwined, leading to a decrease in the perceived size of spatial regions. Memory access planning can only recover a small portion of the original region due to buffering limitations. To address this issue, the authors propose using OS-controlled coloring for bank allocation, where multiple independent banks are assigned to each potentially collision-prone thread. This approach aims to compensate for the lower bank parallelism available to each thread. Additionally, the authors introduce DRAM sub-positioning to effectively increase the number of banks in the system without incurring additional costs. By combining bank partitioning and sub-positioning, the authors aim to enhance production, execution, and affordability.

The study demonstrates that this integrated bank division and sub-positioning strategy can expand the memory region, leading to improvements in efficiency and performance. The effectiveness of this approach is particularly evident in system configurations where each thread is typically allocated a small number of banks. Considering the cost implications of expanding memory banks versus increasing the number of simultaneous threads in the CPU, the authors suggest that future systems should adopt balanced bank-oriented designs. Overall, the proposed bank partitioning and sub-positioning technique offers a promising solution for optimizing memory architectures. By leveraging this approach, systems can achieve increased efficiency and performance, especially in scenarios where applications within a multi-threaded workload have a wide geometric region.

3. Implementation

The server module plays a crucial role in this application as it handles the overall management of customer activities. Upon receiving data from customers, the server performs thorough monitoring and examination. It securely stores the files uploaded by the customers while also maintaining a comprehensive record of all the files uploaded by different clients. The server acts as the central control unit, ensuring the smooth operation and secure data storage. It effectively manages and processes the data for efficient file matching and organization.

The Client module represents the end user of the application. Initially, the client is required to register by providing the necessary details. Once registered, the client can log in to the server using a unique username and password. Upon successful login, the client gains access to upload files to the server, which undergoes security verification. During the upload process, the server performs file matching to identify if the same data has been previously uploaded or if the file is distinct from existing files stored on the server.

In order to partition memory banks effectively, it is important to understand the memory requirements of each component. These requirements are primarily determined by three factors: memory intensity, locality of memory accesses, and bank-level parallelism. Memory intensity refers to the frequency at which an application generates memory requests. It indicates how often the application accesses memory. The concept of locality of memory accesses pertains to the spatial and temporal patterns exhibited by memory requests. It characterizes the degree of proximity and timing between related memory requests. By analyzing the locality of memory accesses, we can gain insights into the patterns and behaviors of memory operations, which can inform optimization strategies for improved performance and efficiency. Bank-level parallelism is a crucial factor that influences the efficiency and performance of memory access operations. It refers to the capability of accessing multiple memory banks simultaneously. By leveraging bank-level parallelism, the system can optimize memory access and improve overall speed and efficiency.

Record coordination serves as the core function of this application. Initially, during data upload, the data undergoes a preprocessing step utilizing the SHA1 hashing system. Each data is assigned a hash key value. The server then performs a check on both the file and its attributes using a genetic approach that applies fitness functions. AES encryption algorithm is applied to encrypt all the files. The server performs matching by comparing the properties of the uploaded files with existing documents or by matching the filenames alone. This process enables the detection and avoidance of duplication, ensuring efficient coordination of records. Based on

the outcome of the record coordination process, if a new file being uploaded has the same file quality as an existing document and shares the same file name, a message indicating that the file already exists will be displayed. In this case, a duplicate file will be created and stored in a separate server. On the other hand, if the file name is the same, but the file quality is different, the file name will be updated and then stored in the server. By effectively identifying file coordination and duplicate identification, the server's efficiency is enhanced, and unnecessary memory usage is minimized. In the Mapping Technique, the primary focus is on minimizing server storage capacity. This is achieved by identifying and deduplicating records, emphasizing the client-level approach. In a cloud server environment, only one copy of a file is stored for all clients who have the same data. The MPT (Memory Mapping Technique) Technique is employed to enable all clients to access the same file efficiently.

In a simplified explanation, one can encrypt or decrypt the plaintext by performing a selective XOR operation with a key. This step can be easily reversed by performing another selective XOR operation with the same key on the ciphertext. In the case of the AES , multiple rounds are used, each requiring its own key. The actual key is "extended" and transformed to provide key segments for each round. This process is known as key expansion, which is the focus of this section. The key expansion schedule, as part of the overall AES algorithm, takes an input key (referred to as the specified key) of 4Nk bytes or Nk 32-bit words. The value of Nk can be either 4, 6, or 8. The output is an extended key (referred to as w) of 4Nb*(Nr+1) bytes, where Nb is always 4 and Nr is the number of rounds in the algorithm. If Nk is 4, Nr is equal to 10. If Nk is 6, Nr is equal to 12. If Nk is 8, Nr is equal to 14. The key expansion routine described below defines most of the operations in terms of words or 4-byte units, as the AES specification itself emphasizes words. However, my implementation uses bytes exclusively.

In summary, the exchange of things in the above pseudo-code can be described as follows:

- The constant Nb = 4: This value represents the number of words in an AES block and is consistently set to 4 throughout the algorithm.
- The key, key: The input key used for encryption or decryption consists of Nk words, which is equivalent to 4*Nk bytes.
- The extended key, w: This key is generated through the key expansion process and is used in subsequent rounds of the AES algorithm. The size of the extended key is determined by Nb(Nk+1) words or 4Nb*(Nk+1) bytes.

The table below provides the range of possible sizes for the extended key, based on different values of Nk:

Table 1:

Nk value	Nr value	Extended Key Size (in words)
4	10	44
6	12	52
8	14	60

These sizes indicate the total number of words in the extended key for respective values of Nk and Nr.

The RotWord() function performs a simple cyclic shift operation on a word, transforming [a0,a1,a2,a3] to [a1,a2,a3,a0]. The Rcon[i] is defined as the word [xi-1,0,0,0]. The table below provides the values of powers of x:

Table 2:

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
xi	1	2	4	8	10	20	40	80	1b	36	6c	d8	ab	4d	9a

Powers of x = 0x02. Please note that the table contains the corresponding values for different powers of x.

In the calculation for key expansion, it is important to mention that the initial reference to Rcon is Rcon[i/Nk], where i has a value of Nk. This ensures that the smallest index to Rcon is 0, utilizing x0 as the value. The Sub Word() function simply applies the S-box value used in Sub Bytes to each of the 4 bytes in the argument.

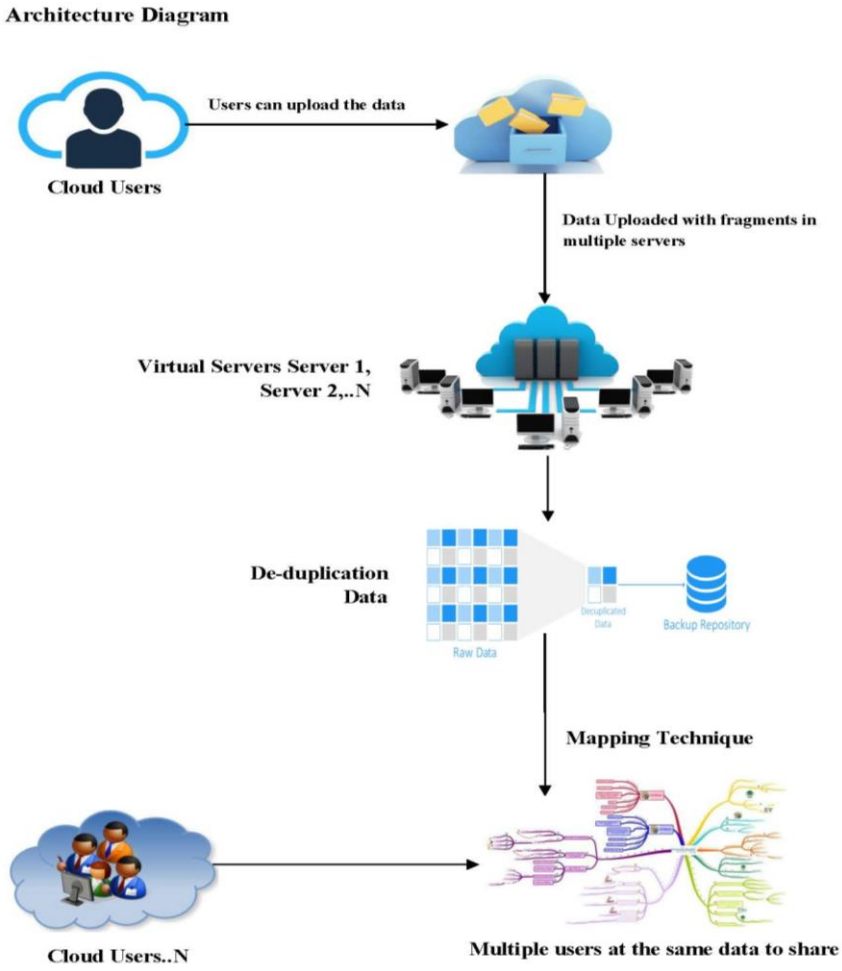

Figure 1:

4. Research Studies

This section provides an overview of some data deduplication research studies that are pertinent to the goals of this work. The protected data deduplication strategies proposed in this work are discussed along with concepts and types of algorithms that are pertinent to them.

4.1 Framework for Secure Data Deduplication in Cloud Environments

Data deduplication techniques commonly involve dividing the data, such as a file, into smaller units known as chunks. Various chunking methods have been proposed in the literature [6]. In the Secure Data Deduplication framework, we recommend using the Two Thresholds Two Divisors (TTTD) method for file division [7]. This method is particularly effective in handling small insertions and deletions within a file. According to Kave and Tang [7], using the TTTD approach only affects the surrounding chunks when there are changes to the file content. It is important to note that the TTTD approach was initially used in data backup servers and has not been widely implemented in cloud environments.

The Content Defined Chunking (CDC) algorithm, which is a specialization of the TTTD algorithm, employs the Basic Sliding Window (BSW) technique [8]. In CDC, a chunk boundary is established if the distance from a reference point, such as the previous boundary chunk or the beginning of the file, reaches a predefined

threshold value denoted as "t". By allowing these algorithms to identify unchanged portions of the data, the deduplication process can be made more efficient. However, it is possible for these methods to occasionally result in chunk sizes that are either too large or too small.

To address the issue of chunk sizes in the TTTD algorithm, it incorporates a lower bound threshold (Tmin) to discard chunks below this threshold and breaks down portions exceeding an upper bound threshold (Tmax) into fixed-size chunks. The TTTD algorithm has shown high effectiveness in chunking both simulated and real-world big data files [6]. In our Secure Data Deduplication framework, to ensure the privacy of the data, each file chunk is secured using the convergent encryption approach [9]. Unlike using a user-selected random cryptographic key, convergent encryption utilizes the actual content of the data to generate the same ciphertext at each iteration. This characteristic makes it suitable for data deduplication strategies that prioritize user privacy standards. It is important to note that symmetric key and public key encryption techniques require the creation of unique keys by or for different users, resulting in different ciphertexts for the same plaintext. Hence, they are not suitable for cross-user data deduplication.

The convergent encryption technique developed in [9] was initially intended for disk-based data deduplication systems and not specifically designed for cloud storage environments that require efficient search and retrieval methods. In the paper [10], it is suggested that combining the convergent encryption algorithm with a data deduplication scheme could potentially enable infinite storage service capability, assuming that the cloud storage providers are trusted and provide full access to the users' data. However, there is no concrete proof to support this claim.

In addition to traditional methods, machine learning-based techniques have been proposed for indexing and retrieving data in deduplication systems. Dinerstein et al. [11] introduced a strategy based on Support Vector Machines (SVM) to develop deduplication rules. However, their approach heavily relies on specific criteria and data gathering, making it less suitable for a generic cloud storage environment. Another alternative method suggested by Benjamin et al. [12] is the use of Bloom filters to determine whether a piece of data is new to the system. Similar to Lillibridge et al. [13], they make use of the same localization assumption for streamed data. One of the core elements of our proposed solution is the encryption of data indices using an asymmetric searchable encryption technique [14]. The idea of using asymmetric searchable encryption, along with symmetric or multi-user schemes, to protect cloud users' data from unreliable cloud providers was initially introduced in [14]. In cloud storage systems, it is common for different users to write and search for data, or for there to be multiple readers but a single creator of the data. Our proposed method allows users to outsource or write the data in the cloud while enabling the cloud service provider (CSP) to search or read the data using an asymmetric searchable encryption scheme.

While the single writer and reader configuration may not always be supported in data deduplication scenarios, it aligns with symmetric searchable encryption algorithms. Multi-user symmetric searchable encryption approaches work when there are multiple readers and only one writer for the data, which is similar to the context of data deduplication in the cloud. However, since the CSP is the only reader in the cloud environment, this configuration may not be suitable for data deduplication. Researchers have explored conjunctive searches and various techniques for executing complex search queries in the context of asymmetric searchable encryption schemes 15. Additionally, there have been investigations into applying these encryption algorithms to protect the privacy of the responses 17[19] and addressing implementation challenges in real systems.

The final module of our architecture focuses on utilizing proof of storage to verify the integrity of data stored in the cloud. The concept of retrievability proof for large files is discussed in [20]. Ateniese et al. proposed a model for storing verifiable data on untrusted servers in [21]. Their strategy reduces network traffic and interaction with the server, as the user does not need to retrieve the entire dataset to verify its integrity. Instead, they sample a small selection of data sets. To the best of our knowledge, these approaches have not yet been extensively applied in the context of cloud computing or data deduplication scenarios.

4.2 Secure Enterprise Data Deduplication in the Cloud

Secure enterprise data deduplication in the cloud involves the use of methods to effectively remove redundant data while prioritizing the privacy and security of sensitive information. This process includes breaking files into smaller chunks, employing encryption for data privacy, utilizing machine learning techniques for

indexing and retrieval, and integrating proof of storage mechanisms to ensure data integrity. By implementing these strategies, organizations can achieve efficient and reliable data deduplication within cloud storage environments while safeguarding their data. In the initial step of the Secure Enterprise Data Deduplication Scheme, the file chunking component plays a crucial role in dividing a given data file into multiple pieces of varying sizes. To maintain an updated chunk index, it is recommended to utilize B trees, specifically B+ trees, which are commonly used in disk-based systems [22].

Different indexing approaches have been proposed in the literature to ensure the security of the indexing process. For instance, according to [23], B+ trees are efficient data structures that can be employed to construct a secure index for a database. A proof of concept is presented, suggesting the organization of nodes into buckets and the application of homomorphic encryption on these groups to conceal access patterns from potential attackers. However, this technique does not consider the use of homomorphic encryption within the database, which is essential for achieving consistent ciphertext for the same plaintext, a crucial aspect of data deduplication in our approach. Instead, it focuses on protecting the access and search patterns of the cloud service provider (CSP).

In [24], a data chunking method is used to divide the data into multiple parts, forming the nodes of the B+ trees used for indexing cloud-based data. This approach establishes a local index and a broader index based on structured overlay, aiming to reduce the amount of data transmitted to the cloud and accelerate the development of database back-end applications. However, it overlooks security concerns and the implementation of data deduplication within the database. Curtmola et al. introduced the concept of searchable encryption [25], which utilizes cryptographic primitives to enable keyword-based searches on an encrypted database without disclosing the keywords to the CSP. Similarly, a similar technique known as private keyword search was proposed in [26]. Boneh et al. introduced the concept of an initial public key searchable encryption [27], where users (rather than the data owner) can search the encrypted data using public and private keys. Additionally, [28] identifies database queries that utilize identity-based encryption and hash chains for integrity protection, along with an audit log. These various encryption-based techniques provide options for secure and privacy-preserving search operations on encrypted data, contributing to the overall security and effectiveness of enterprise data deduplication in the cloud.

4.3 Proof of Retrieval and Ownership Protocols for Data Deduplication

In the context of cloud or distributed storage systems, proof of retrieval and ownership protocols are vital for safeguarding the integrity and security of data in cross-user scenarios. These protocols establish mechanisms to verify the retrievability and ownership of stored data. By implementing these protocols, organizations can ensure that their data remains protected and can be successfully retrieved when needed. This is especially important in scenarios where multiple users or entities are involved and share the same storage infrastructure. The protocols contribute to upholding data integrity, preventing unauthorized access, and facilitating efficient retrieval of data. Overall, the implementation of these protocols enhances the overall security and reliability of data deduplication processes in cloud or distributed storage environments.

Proof of Retrieval (POR) is a concise statement provided by a storage provider to a client, assuring that the target file F remains intact and can be fully recovered by the customer [29]. This concept was introduced by Juels and Kaliski [29]. Utilizing Proof of Data Possession (PDP) techniques allows users to verify the reliability of data stored in the cloud without necessarily guaranteeing the ability to recover the entire file. Ateniese initially proposed this concept [30], which was later developed in subsequent works [31]. Proof of Ownership (POW) protocols provide assurance to the Cloud Service Provider (CSP) that the security of the data has not been compromised by unauthorized access [33].

One of the earliest studies examining the combination of POR mechanisms with POW and PDP procedures is discussed in [84]. This research describes a distributed setup where a file is divided into segments and assigned to different servers. These servers verify the integrity of the data by comparing the Message Authentication Codes (MACs) of the assigned blocks with each other to ensure data accuracy. However, this work lacks explicit definitions of the proposed architectures and does not include a comprehensive security analysis. In [30], a valuable contribution is made in the field of PDP with a proof-of-data-possession system that integrates file blocks and homomorphic verifiable tags. This system is accompanied by a security proof based on the concept

of game theory. However, these schemes have two drawbacks: they are impractical for real-world applications due to their requirements, and they prohibit data retrieval in case of corruption. To address these limitations, later works [31] present a publicly verifiable version of these techniques, guaranteeing an infinite number of responses between the user and the server.

These existing approaches can be further improved to minimize overhead, which remains nearly constant regardless of the size of server-side files, as proposed in [32]. However, the feasibility of these enhanced approaches for a wide range of applications is still being evaluated. It's important to note that these approaches have limitations in terms of data retrievability when data corruption occurs.

Another relevant work by Shachams [35] suggests the use of homomorphic authenticator tags in file blocks. This strategy involves averaging the tag values over multiple blocks, reducing the bandwidth requirement. The approach also allows for an unlimited number of trials. However, none of the aforementioned techniques specifically address data deduplication, which is an important consideration in storage systems. The concept of data sentinels was introduced in [36], where a POR protocol was established using error-correcting coding methods. Sentinels, which are error-correcting code checks, are generated independently of the bit strings being validated for retrievability [36]. These sentinels are randomly scattered throughout the stored data. An enhancement to this idea is the twofold encoding of data, which provides a comprehensive Byzantine adversarial approach [29]. However, since sentinels are inserted into the original data blocks, these methods are not suitable for data deduplication. This is because the randomly placed sentinels make it challenging for the CSP to identify and reject similar data. In summary, while these approaches show promise in improving efficiency and reliability, there is ongoing research to evaluate their feasibility in various applications. Additionally, considerations for data deduplication need to be addressed to further enhance the effectiveness of these techniques.

A proof-of-storage (POS) approach that facilitates data deduplication is proposed in [37]. In this approach, the data is divided into distinct "chunks," and deduplication tags are generated for each chunk of the file. Both the data and these tags are then transferred to the cloud for further authentication. It is important to note that in this approach, the Cloud Service Provider (CSP) is considered a fully trusted party as it has access to the unencrypted information. The security of this approach is evaluated using a Random Oracle Model based on computational Diffie-Hellman assumptions. In the context of proof-of-retrieval (POR) protocols, [38] and [39] consider data activities at the block level. [38] employs the Sobol random sequencing approach, but it is not suitable for data deduplication as the random arrangement of blocks makes it challenging for the CSP to detect duplicate data. On the other hand, [39] proposes a POR technique for dynamic data using homomorphic tags and Merkle Hash Trees. However, as mentioned in [32], the usefulness of the chunk tags is reduced as they are calculated and uploaded. Regarding data deduplication, proof-of-ownership (POW) approaches are presented in [33] and [40]. These methods assume a reliable CSP and utilize the Merkle Hash Tree method to establish proof of ownership. This paper enhances the POR approach from [36] and the POW approach from [33] to accommodate the requirements of data deduplication for textual, picture, and video data, assuming that the CSP is semi-honest. As far as our knowledge extends, our POR and POW approaches are unique in transforming data compression, data deduplication, and proof of storage algorithms in scenarios where the CSP is considered semi-honest.

5. Results And Disscussion

Per-file parity (PFP) is introduced to address the need for simultaneously ensuring individual documents while providing higher-level protection for critical data pieces in deduplication-based storage systems. The implementation of PFP offers valuable insights and highlights that data deduplication has a dual impact on system reliability. Deduplication is commonly employed in backup and archival scenarios to reduce the backup window and save storage space significantly. It reduces the storage footprint, lowers the cost, and minimizes the number of storage devices, thereby reducing the risk of data loss. Additionally, data deduplication greatly enhances the reliability of flash-based storage systems by reducing write traffic to the devices. This reduction in writes helps minimize internal write operations and garbage collection (GC) tasks, which can negatively impact the reliability and performance of flash devices. Therefore, it becomes essential to conduct reliability analysis for deduplication-enabled HDD-based and SSD-based storage systems.

PFP is introduced to provide both individual document protection and higher-level data protection in deduplication-based storage systems. Data deduplication offers advantages such as shorter backup windows,

reduced storage costs, and improved reliability for flash-based storage systems. However, the impact of deduplication on system reliability is very complex and needs a comprehensive analysis of various storage technologies. There are two key considerations for achieving higher reliability in storage systems: the use of erasure codes and the management of fingerprint data and metadata in deduplication-based systems.

Firstly, erasure codes can be applied to enhance reliability by allowing for the recovery of data in the presence of multiple failures. With the increasing processing power of CPUs in host machines and device controllers, erasure codes have become widely used in storage systems with manageable processing overhead. In addition to the commonly used parity and replication redundancy in RAID-based systems, erasure codes can tolerate two or more data chunk failures within a code group or code word. Therefore, incorporating erasure codes into the Per-File Parity (PFP) scheme is an avenue for future work. This integration can provide higher levels of data protection and prevent file losses caused by simultaneous failures of multiple data chunks within a code group.

Secondly, the management of fingerprint data and other metadata is crucial in deduplication-based storage systems. In the PFP scheme, a separate mapping table is necessary to manage files that share high-reference-count data chunks. This mapping table enables inter-file recovery when intra-file recovery is not sufficient to recover lost data chunks. While the inter-file recovery module can be integrated into the background disk cleaning process to reduce metadata overhead, it may impact system performance. Incorporating erasure codes into the PFP scheme can enhance reliability by tolerating multiple data chunk failures. Additionally, efficient management of fingerprint data and metadata is essential to support inter-file recovery in deduplication-based storage systems.

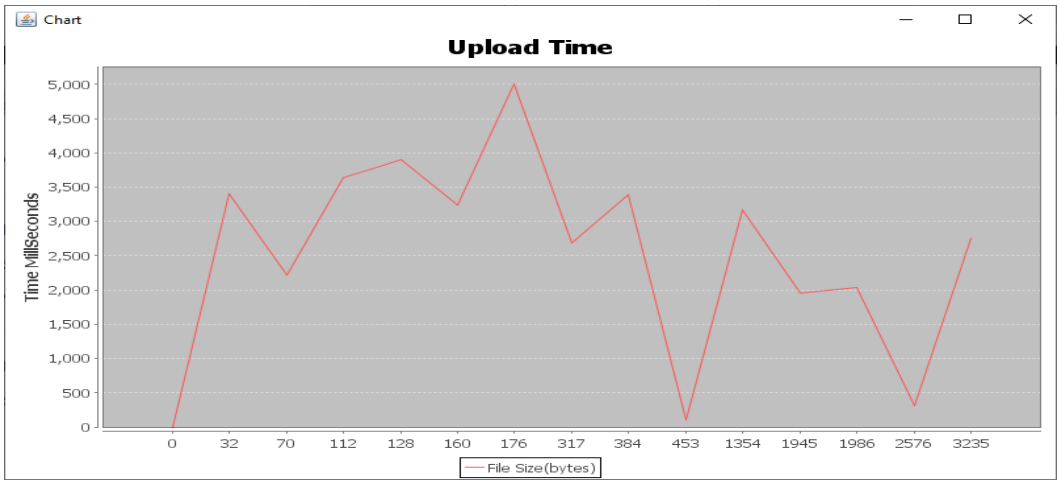

Figure 2:

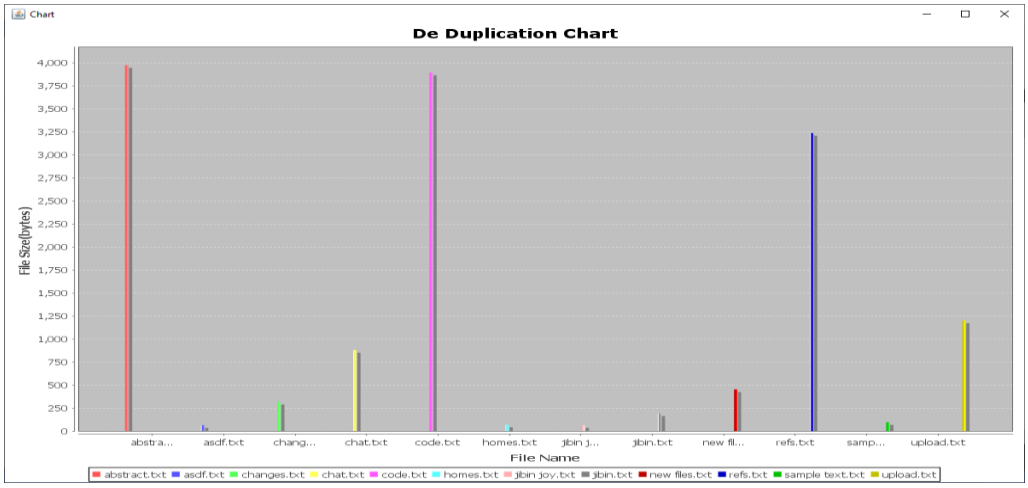

Figure 3:

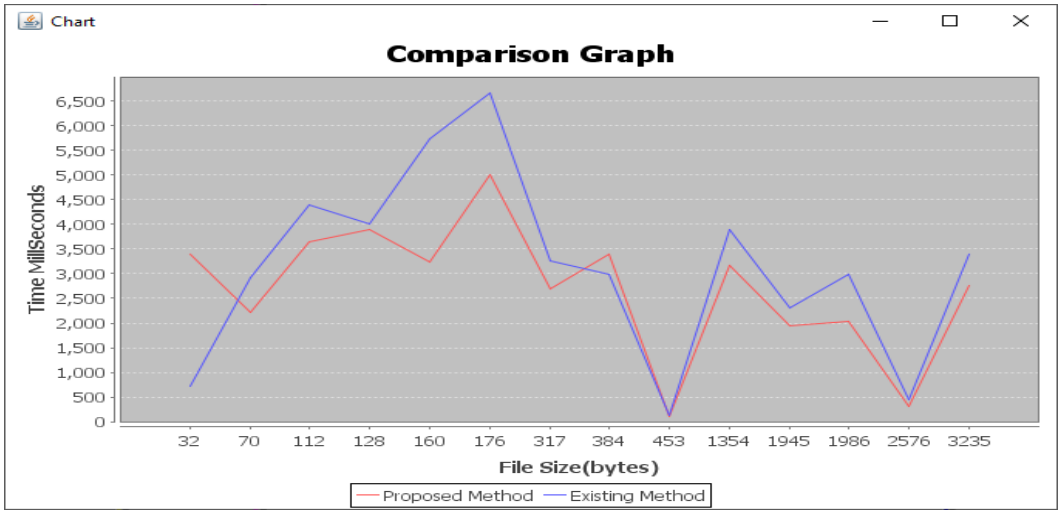

Figure 4:

6. Conclusion And Future Work

As the digital data usage continues to increase exponentially, cloud storage services have gained popularity due to their promise of convenient and efficient storage solutions that can be accessed anytime, anywhere. The advent of cloud computing and its related storage providing services has addressed the challenge of managing the overwhelming amount of electronic data at both individual and business levels. Cloud technology provides a platform for storing, processing, and accessing large volumes of data, meeting the demand for accessible storage solutions. However, despite the advantages of cloud storage, concerns about data privacy have limited user acceptance. While data is typically encrypted when outsourced to the cloud, the confidentiality of sensitive data is not guaranteed when a trusted but observant Cloud Service Provider (CSP) controls the storage. This raises concerns about the security of data stored in the cloud, particularly when it comes to privacy.

Integrating data deduplication into cloud storage can enhance the amount of data that can be stored by reducing data redundancy and replication. Data deduplication ensures that only one copy of previously duplicated data is maintained, optimizing storage space for CSPs. However, this approach also introduces significant security and privacy risks for users. The process of data deduplication involves analyzing and comparing data chunks, which can potentially expose sensitive information to unauthorized access or breaches. Cloud storage services offer convenient and scalable solutions for managing large amounts of data, concerns remain regarding data privacy and security. The integration of data deduplication can improve storage efficiency for CSPs but poses challenges in maintaining user privacy and data security. Balancing the benefits of cloud storage with the need for data confidentiality is a crucial consideration in the adoption of cloud technology. We have developed a two-level data deduplication framework that enables businesses utilizing the same CSP's data storage services to optimize space savings and reduce costs. The framework consists of cross-user level deduplication and cross-enterprise level deduplication, allowing the CSP to efficiently handle data deduplication at both the organizational and single-user levels.

To ensure security, we have implemented a secure indexing approach using B-trees. This index is used to manage data deduplication and is encrypted using convergent encryption to enhance data security. At the organizational level, we have employed a private keyword search approach, enabling multi-user searchable encryption. This approach facilitates secure and practical internal file sharing, which is essential for effective business operations.

In the context of cross-user client-side data deduplication for medium- and small-sized organizations, we have established a privacy-preserving technique to ensure data integrity. This technique includes proof of retrievability (POR) and evidence of ownership (POW) protocols that validate the integrity of data stored in the cloud. Through a comprehensive security study, we have demonstrated that our framework is secure against both internal and external attacks. It safeguards against attacks that attempt to identify files, access their contents, or exploit vulnerabilities. Furthermore, our analysis indicates that the POR and POW protocols allow users to securely query the CSP while preventing malicious users from compromising data integrity. In terms of

performance, our framework efficiently distributes the computational load across users and servers, resulting in low operational costs. This is achieved by leveraging previously generated data during various operations. Overall, our two-level data deduplication framework offers businesses a secure and cost-effective solution for optimizing storage space and managing data deduplication within the cloud storage environment.

In order to enhance the security of our recommended deduplication strategies, we have plans to develop a future technique based on compressed sensing (CS). This approach will utilize different measurement matrices and sampling techniques for various types of data, such as text, video, and images, to reduce computational costs while ensuring the security of the deduplication systems. We aim to explore Non-Deterministic and Non-Adaptive Measurement matrices/encodings, as well as Non-Deterministic and Adaptive Measurement matrices/encodings, specifically for deduplication purposes. By considering these different forms of data, we can optimize the deduplication process and enhance its effectiveness. In addition to data security, we also intend to provide privacy-preserving public auditing of uploaded data in the cloud. Our goal is to prevent the exposure of data content to third-party external auditors (TPAs) while allowing them to verify the accuracy of the data. This will be achieved by leveraging the concept of homomorphic linear authenticators (HLAs), which eliminate the need for the auditor to download the entire dataset from the cloud.

To complement the HLA solutions, we plan to incorporate fundamental Message Authentication Code (MAC) solutions for this problem. These additional security measures will further strengthen the integrity and authenticity of the deduplicated data. Furthermore, we are exploring the use of artificial intelligence (AI) approaches, including neural networks, to efficiently and rapidly identify duplicates. By leveraging AI techniques, we can develop more advanced and cost-effective methods for detecting duplication, improving the overall deduplication process. Finally, we are also preparing to address the scenario where dynamic data resides in the cloud and a Cloud Service Provider (CSP) needs to perform data deduplication to optimize digital space while maintaining data security. This will ensure that the deduplication process remains effective and robust even in dynamic cloud environments. Overall, these future developments aim to reinforce the security, efficiency, and effectiveness of deduplication strategies, providing businesses with advanced techniques to manage their data and optimize storage resources.

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] C. Bo, Z. F. Li, and W. Can, \Research on chunking algorithm of data deduplication,"American Journal of Engineering and Technology Research, vol. 11, no. 9, pp. 1353 {1358, 2011.

[7] E. Kave and H. K. Tang, \A framework for analyzing and improving content based chunking algorithms," tech. rep., International Enterprise Technologies Laboratory, HP Laboratories Palo Alto, Sept 2005.

[8] A. Muthitacharoen, B. Chen, and D. Mazieres, \A low-bandwidth network le system," in Proceedings of the eighteenth ACM symposium on Operating systems principles, SOSP '01, (New York, NY, USA), pp. 174{187, ACM, 2001.

[9] C. Wang, Z. guang Qin, J. Peng, and J. Wang, \A novel encryption scheme for data deduplication system," in Communications, Circuits and Systems (ICCCAS), 2010

[10] International Conference on, pp. 265 {269, july 2010. [20] S. Parez, \Bitcasa: Innite cloud storage." http://techcrunch.com/2011/09/18/bitcasa- explains-encryption/, Sept 2011. Online.

[11] J. Dinerstein, S. Dinerstein, P. Egbert, and S. Clyde, "Learning-based fusion for data deduplication," in Machine Learning and Applications, 2008. ICMLA '08. Seventh International Conference on, pp. 66 – 71, dec. 2008.

[12] B. Zhu, K. Li, and H. Patterson, "Avoiding the disk bottleneck in the data domain deduplication file system," in Proceedings of the 6th USENIX Conference on File and Storage Technologies, FAST'08, (Berkeley, CA, USA), pp. 18:1–18:14, USENIX Asso- ciation, 2008.

[13] M. Lillibridge, K. Eshghi, D. Bhagwat, V. Deolalikar, G. Trezise, and P. Camble, "Sparse indexing: large scale, inline deduplication using sampling and locality," in Proccedings of the 7th conference on File and storage technologies, FAST '09, (Berkeley, CA, USA), pp. 111–123, USENIX Association, 2009.

[14] S. Kamara and K. Lauter, "Cryptographic cloud storage," in Proceedings of the 14th international conference on Financial cryptograpy and data security, FC'10, (Berlin, Heidelberg), pp. 136–149, Springer-Verlag, 2010.

[15] D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field key- word search," in Proceedings of the 5th international conference on Information Secu- rity Applications, WISA'04, (Berlin, Heidelberg), pp. 73–86, Springer-Verlag, 2005.

[16] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in Proceedings of the 4th conference on Theory of cryptography, TCC'07, (Berlin, Hei- delberg), pp. 535–554, Springer-Verlag, 2007.

[17] J. Baek, R. Safavi-Naini, and W. Susilo, "On the integration of public key data encryp- tion and public key encryption with keyword search," in Proceedings of the 9th interna- tional conference on Information Security, ISC'06, (Berlin, Heidelberg), pp. 217–232, Springer-Verlag, 2006.

[18] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," in Proceeding sof the international conference on Computational Science and Its Applications, Part I, ICCSA '08, (Berlin, Heidelberg), pp. 1249–1259, Springer- Verlag, 2008.

[19] T. Fuhr and P. Paillier, "Decryptable searchable encryption," in Proceedings of the 1st international conference on Provable security, ProvSec'07, (Berlin, Heidelberg), pp. 228–236, Springer-Verlag, 2007.

[20] A. Juels and J. Burton S. Kaliski, "Pors: proofs of retrievability for large files," in Proceedings of the 14th ACM conference on Computer and communications security, CCS '07, (New York, NY, USA), pp. 584–597, ACM, 2007.

[21] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proceedings of the 14th ACM con- ference on Computer and communications security, CCS '07, (New York, NY, USA), pp. 598–609, ACM, 2007.

[22] T. Thwel and N. Thein, "An efficient indexing mechanism for data deduplication," in Current Trends in Information Technology (CTIT), 2009 International Conference on the, pp. 1 –5, dec. 2009.

[23] H. Pang, J. Zhang, and K. Mouratidis, "Enhancing access privacy of range retrievals over B+trees," Knowledge and Data Engineering, IEEE, pp. 99–99, 2012.

[24] S. Wu, D. Jiang, B. Ooi, and K. Wu, "Efficient B+tree based indexing for cloud data processing," The Proceedings of the VLDB Endowment (PVLDB), vol. 3, pp. 1207– 1218, Sep 2010.

[25] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryp- tion: improved definitions and efficient constructions," in Proceedings of the 13th ACM conference on Computer and communications security, pp. 79–88, ACM, 2006.

[26] Y. Yang, H. Lu, and J. Weng, "Multi-user private keyword search for cloud com- puting," in Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on, pp. 264 –271, 29 2011-dec. 1 2011.

[27] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryp- tion with keyword search," in Advances in Cryptology-Eurocrypt 2004, pp. 506–522, Springer, 2004.

[28] B. Waters, D. Balfanz, G. Durfee, and D. Smetters, "Building an encrypted and search- able audit log," in Proceedings of 11th Annual Network and Distributed System Security Symposium (NDSS 2004), vol. 6, 2004.

[29] K.D.Bowers, A.Juels, and A.Oprea, "Proofs of retrievability: theory and implementa- tion," in Proceedings of the 2009 ACM workshop on Cloud computing security, CCSW '09, pp. 43–54, 2009.
[30] G.Ateniese, R.Burns, R.Curtmola, J.Herring, L.Kissner, Z.Peterson, and D.Song, "Provable data possession at untrusted stores," in Proceedings of the 14th ACM con- ference on Computer and communications security, CCS '07, pp. 598–609, 2007.
[31] G.Ateniese, S.Kamara, and J.Katz, "Proofs of storage from homomorphic identifi- cation protocols," in Proceedings of the 15th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, ASI- ACRYPT '09, pp. 319–333, 2009.
[32] G.Ateniese, R.Burns, R.Curtmola, J.Herring, O.Khan, L.Kissner, P.Zachary, and D.Song, "Remote data checking using provable data possession," ACM Trans. Inf. Syst. Secur., pp. 12:1–12:34, June 2011.
[33] S.Halevi, D.Harnik, B.Pinkas, and S.Alexandra, "Proofs of ownership in remote storage systems," in Proceedings of the 18th ACM conference on Computer and communica- tions security, CCS '11, pp. 491–500, 2011.
[34] M.Lillibridge, S.Elnikety, A.Birrell, M.Burrows, and M.Isard, "A cooperative internet backup scheme," in Proceedings of the annual conference on USENIX Annual Technical Conference, pp. 3–3, USENIX Association, 2003.
[35] H.Shacham and B.Waters, "Compact proofs of retrievability," in Advances in Cryptology-ASIACRYPT 2008, pp. 90–107, Springer, 2008.
[36] A.Juels, Jr.Kaliski, and S.Burton, "Pors: proofs of retrievability for large files," in Proceedings of the 14th ACM conference on Computer and communications security, CCS '07, pp. 584–597, 2007.
[37] Q.Zheng and S.Xu, "Secure and efficient proof of storage with deduplication," in Pro- ceedings of the second ACM conference on Data and Application Security and Privacy, CODASPY '12, pp. 1–12, 2012.
[38] S.Kumar and R.Subramanian, "An efficient and secure protocol for ensuring data stor- age security in cloud computing," International Journal of Computer Science, vol. 8, 2011.
[39] Q.Wang, C.Wang, K.Ren, W.Lou, and J.Li, "Enabling public auditability and data dynamics for storage security in cloud computing," Parallel and Distributed Systems, IEEE Transactions on, vol. 22, no. 5, pp. 847–859, 2011.
[40] 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
[41] Memory Latency Reduction via Thread Throttling by H. Cheng, C. Lin, J. Li, and C. Yang 2010, IEEE, DOI 10.1109/MICRO.2010.39)
[42] 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)
[43] Singleton: System-wide Page Deduplication in Virtual Environments Prateek Sharma Purushottam Kulkarni (2012, ResearchGate, 10.1145/2287076.2287081)
[44] Enhancing Operating System Support for Multicore Processors by Using Hardware Performance Monitoring (2009, ResearchGate, DOI:10.1145/1531793.1531803)
[45] 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)
[46] Enhanced Cloud Data Security Using AES Algorithm (2014, IEEE, DOI: 10.1109/JPROC.2013.2287711)
[47] A Framework Based on RSA and AES Encryption Algorithms for Cloud Computing Services (2014, IEEE, DOI: 10.1109/I2C2.2017.8321820)
[48] Enhanced RSA Algorithm with varying Key Sizes for Data Security in Cloud (2017, IEEE, DOI: 10.1109/WCCCT.2016.50)
[49] DROPS: Division and Replication of Data in Cloud for Optimal Performance and Security (2015, IEEE, DOI: 10.1109/TCC.2015.2400460)
[50] Applying Encryption Algorithm for Data Security in Cloud Storage (2016, ResearchGate, Springer, DOI: 10.1007/978-981-287-990-5_12)