Study of Efficient Cloud Storage Architectures for the Security Environment

Ayad Hasan Adhab
Kufa University, Ph.D.in Computer Science and Mathematics College, Iraq
ayadwasit1978@gmail.com
https://orcid.org/0000-0002-7778-6983

Naseer Ali Hussien
Alayen University, Iraq
naseerali@alayen.edu.iq

DOI: http://dx.doi.org/10.31642/JoKMC/2018/100108

Received Dec. 12, 2022. Accepted for publication Mar. 15, 2023

Abstract— *In recent years, cloud computing has gained significant popularity, with data storage emerging as a crucial and valuable aspect of this technology. Cloud data storage refers to transferring data to a remote, up-to-date storage system rather than relying on a local device such as a computer's hard disc. Cloud storage allows users to reduce their hardware and software requirements while benefiting from enhanced convenience and cost-effectiveness. However, security concerns surrounding data access and storage in the cloud often render it unreliable and untrustworthy. Cloud computing delivers dynamically scalable resources over the Internet, offering numerous advantages in terms of cost and usability.*

Nevertheless, cloud computing services must address security challenges when transmitting sensitive information and critical applications to shared and public cloud environments. As cloud environments expand to meet data processing and storage needs, users must carefully consider the potential benefits and drawbacks of these services regarding data security. This study focuses on the primary security concerns in cloud computing systems, examining the fundamental issues associated with data protection in these environments. We present a secure, privacy-preserving architecture for inter-cloud data sharing that employs an encryption/decryption algorithm to safeguard data stored in the cloud against unauthorized access. By providing a robust and reliable solution to address the security challenges inherent in cloud computing, this research contributes to developing safer and more trustworthy cloud-based systems for users.

Keywords— Cloud computing, Cloud storage, Architecture, Security environment, Data security

I. INTRODUCTION

Cloud computing and the Internet of Things (IoT) have emerged as transformative technologies that are revolutionizing how businesses and individuals interact with the digital world. With the vast amount of data generated by IoT devices, artificial intelligence (AI) can be employed to analyze and extract valuable insights, improving operational efficiency and streamlining processes (Lin et al., 2017). Furthermore, cloud systems offer the necessary computational resources and storage capabilities to support the processing and analysis of IoT data, including video data (Chen et al., 2016).

Combining cloud and IoT technologies presents an attractive solution for many organizations; however, some may lack the bandwidth or computational resources to support a purely cloud-based infrastructure. In such cases, enterprises can explore hybrid options that enable a more strategic, cost-effective, and bandwidth-friendly approach to cloud adoption (Qiu et al., 2022).

A significant concern for organizations utilizing cloud storage is the potential misuse of sensitive data by service providers or other third parties (Ahmed and Sarkar, 2020). To address this issue, researchers have proposed various solutions that enable users to encrypt their data, safeguarding it from unauthorized access. Despite the numerous benefits offered by cloud computing and cloud storage, several research challenges still need to be addressed to overcome barriers related to portability, interoperability, storage access, security, cost, and energy efficiency (Dong et al., 2014). Among these challenges, security—particularly concerning user privacy, legal compliance, and trust issues—remains a significant hurdle to the widespread adoption of cloud technologies (Tancock, Pearson, and Charlesworth, 2012).

A. Background of the study

The US National Institute of Standards and Technology (NIST) defines cloud computing as "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or

service provider interaction" (Mell & Grance, 2011). This cloud model promotes accessibility and encompasses three service types, four deployment models, and five essential characteristics (Takabi, Joshi, & Ahn, 2010).
Cloud computing is revolutionizing the design and procurement of business hardware and software, offering users numerous advantages such as cost savings, flexible resource utilization, and easy web access (Marston, Li, Bandyopadhyay, & Ghalsasi, 2013). As cloud computing becomes increasingly critical to the growth and collaboration of businesses of all sizes, concerns about storing sensitive or confidential information in the cloud persist (Sultan, 2011). In response to these concerns, cloud service providers (CSPs) have implemented security measures such as firewalls and virtualization to protect stored data. However, these measures may need to provide more protection due to network vulnerabilities and CSPs' complete control over cloud applications, hardware, and user data (Zhang, Wu, & Liu, 2017).

Numerous reputable organizations and enterprises offer cloud computing environments, including Google Cloud Platform, Amazon Web Services, and Microsoft Azure. Depending on the services provided, these environments can vary in scope and complexity. Cloud computing services are generally categorized into three main types: infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) (Buyya et al., 2009). Combined, these service types give rise to the concept of Everything as a Service (EaaS).

B. Problem Statement

Data storage on cloud servers is essential as we transition to a more technologically advanced future. Daily, enormous amounts of data are posted to the Internet; maintaining and protecting this data is becoming more complex. Data preservation is crucial; in the current corporate age, it is even required by law. Cloud computing is the most efficient technology at our disposal for securing such a large volume of data. Instead of keeping data on a local server, cloud computing involves utilizing a network of distant computers hosted on the Internet to store, manage, and analyze data. Every cloud has a certain amount of storage, so if we start uploading duplicate information, the storage will be lost, and data redundancy will become a major issue. Researchers have investigated numerous techniques to combat this, and data deduplication is the best answer. A method called data deduplication was developed to improve storage. Cloud service providers, including Dropbox, Amazon S3, and Google Drive, now use this strategy. Data duplication is prevented by ensuring it is only uploaded to the cloud once.

C. Significance of the Study

Cloud computing can use a variety of security measures. The control-based technologies that makeup cloud computing operate under policies that are adjusted to adhere to laws, compliances, and rules that safeguard data through their infrastructures. Additionally, the nature of sharing in cloud computing environments raises questions about its identity management, access controls, and privacy policies. From an algorithmic standpoint, it is necessary to review conventional security algorithms used in cloud computing and improve them with novel techniques consistent with current scenarios. The importance of this study lies in its in-depth analysis of cloud data security using the literature on various algorithms and its first-hand innovative method proposal to ensure more secure cloud data.

D. Objectives of the Study

These are the goals for the present study:

- To design a new mechanism to improve the performance of a large storage system by applying the deduplication technique.
- To evaluate the proposed mechanism's performance compared to available solutions in a simulated environment.
- To verify and validate the proposed mechanism based on the results obtained from the simulation experiments that ensure the correctness of its implementation.

II. REVIEW OF LITERATURE

W. Xia et al. in (2016) suggest FastCDC, a Fast and effective CDC approach, which constructs and enhances the latest Gear based on the CDC technique, one of the fastest CDC techniques to our knowledge. FastCDC's main idea is to integrate five key mechanics: gear-rely on rapid rolling hash, improving and simplifying Gear hash (GH) verdict, skipping sub-minimal chunk cut-points, normalizing the chunk-size distribution in a small specific region to address the issue of reduction deduplication ratio caused by cut-point skipping. FastCDC is around ten times quicker than the best open-source Rabin-based on CDC and about three times greater than the state-of-the-art Gear- and AE-rely on CDC, while obtaining almost the same deduplication ratio as the standard Rabin-rely solution, according to our evaluation results.

N. Kumar and S. Jain (2019) suggest Differential Evolution DE-rely on TTTD-P optimized chunking to maximize chunking throughput while increasing the deduplication ratio DR. Using a scalable bucket indexing strategy minimizes the time it takes to find and declare duplicated hash values (HV). It chunks about 16 times greater than Rabin CDC, five times greater than AE CDC, and 1.6 times greater than FAST CDC (HDFS).
M. Ellappan and S. Abirami (2021) suggest a novel chunking algorithm called Dynamic Prime Chunking (DPC). DPC's major purpose is to dynamically modify the window size during the prime value, relying on the maximum and minimal chunk size. DPC in the deduplication scheme gives good throughput while avoiding large chunk variance. The

multimedia and operating system datasets were used for implementation and experimental evaluation. Existing algorithms such as AE, MAXP, TTTD, and Rabin have been compared to DPC. The performance indicators were throughput, chunk count, Bytes Saved per Second (BSPS), chunking time, processing time, and Deduplication removal Ratio (DER). BSPS and throughput have both improved. To begin, DPC boosts throughput performance by greater than 21% compared to AE. BSPS improves performance by up to 11% over the previous AE method.

P. Anitha et al. in 2021: The secure authorities are given access control mechanisms to do data deduplication (DD) on the outsourced data. Encryption techniques are used in the access control mechanism. It employs convergent randomization and reliable distribution of owning party keys to allow the cloud service provider to manage outsourced data access even when control shifts on a regular basis. The suggested technique for safeguarding data integrity against attacks relies on label discrepancies. As a precaution, the suggested technique has been changed to improve security. By combining three methods, Xu and W. Zhang 2021 describe how QuickCDC improves CDC chunking speed, deduplication ratio, and throughput. Initially, QuickCDC can instantly move to the chunk boundaries of frequently arising duplicate chunks. The mapping of the duplicate chunk's first n bytes and last m bytes to chunk length must be registered. The current chunk's first n bytes and last m bytes are checked to see if they are in the mapping table when Chunking is performed. QuickCDC can skip relevant chunk lengths (CL) if they are in the mapping table. QuickCDC can skip the minimal chunk length for unique chunks. Finally, QuickCDC may dynamically alter mask bits length such that chunk length (CL) is permanently greater than the minimal chunk length and is distributed in a particular limited location. When the current chunk length (CL) is less than the expected chunk length (CL), we should use longer mask bits, and when the current chunk length (CL) is more than the expected chunk length (CL), we should use shorter mask bits. Experiments show that QuickCDC's chunking speed is 11.4x that of RapidCDC, and the associated deduplication ratio is somewhat increased, with a maximum deduplication ratio improvement of 222.3% and a throughput improvement of 111.4%.

III. PROPOSED METHODOLOGY

A. The Proposed System

Our goal is to reduce the computational cost by comparing current work to earlier Content Defined Chunking **(**CDC) methods. We thus suggest the Dynamic Prime Chunking method (DPC), which lowers the computational cost and increases chunking performance, to address the shortcomings of current approaches. Similar to AE, DPC employs variable-size windows as opposed to fixed-size windows. The DPC uses two separate windows—variable size and dynamic variable size windows. The DPC method has no retracing principles to calculate the extreme or maximum value. Two conditional branches and one comparison operation are used to scan the bytes. DPC sets minimum and maximum chunk size restrictions and greatly lowers low entropy bytes produced by other CDC methods. Following are the contributions we made to this study:

(1) We suggest DPC, which lowers chunk duplication in cloud storage by dynamically varying the window size inside the prime number depending on the minimum and maximum threshold using a hash less CDC method.

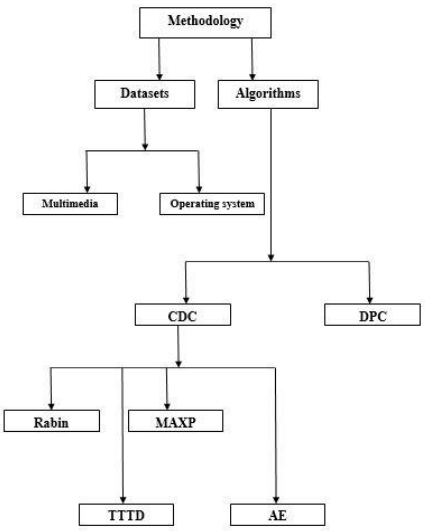
(2) Illustrations and comparisons with different CDC methods are provided for our experimental assessment of the suggested approach for multimedia and operating system datasets.

(3) Using other CDC methods, our DPC analyses the various performance parameters, including chunking count, processing time, chunking time, throughput, BSPS, and DER.

(4) According to the above performance metrics analysis, our suggested technique increases the deduplication system's overall throughput and processing speed.

(5) Compared to other CDC methods already in use, our suggested DPC approach lowers the computational cost.

Figure: 1. Proposed datasets and algorithms



B. Dataset

There are mainly two datasets that will be used, which are as follows

- Multimedia: Personal files for the home folder and media files will be included in the datasets for our research. The files will be removed from the personnel PC, research workstation, and department lab resources. Movies, sports videos, videos from online course materials, and video surveillance are the major types of media datasets

that will be kept up to date by the Department server. MP4, AVI, MKV, and more video file types.

- Operating system: The operating datasets will include several software-related compressed files and the many different versions of the Linux operating system images. The datasets will have a greater degree of content similarity with one another. Table 1 presents the information on the dataset for perusal.

Table 1: Multimedia and OS Datasets

Type	Size (GB)	No. of Files
Home	46.3	9010
Media Files	29.3	259
Linux Files	35.2	16
Compressed	32.3	44

C. S simulation Used

A study has been implemented that focuses on the encryption algorithms implemented by these researchers (mentioned in the literature review). The strength and weaknesses of various algorithms that are utilized in IoT security are analyzed. Most of the work has been done on the Signcryption algorithm, ECC algorithm, and DYNAMIC PRIME CHUNKING algorithm. Signcryption is an encryption technique with the capability and functionality of performing mutually digital signatures and encryption in a single, consistent step. The amalgamation of Signcryption with ECC proves to be outstanding regarding various security parameters.

D. Algorithms used

The key component of data deduplication is the CDC technique, which divides the incoming data stream into manageable blocks or chunks and checks that the data content window before the cut-point fulfills a predefined criterion. The CDC technique also pinpoints the duplication chunks, impacted deduplication ratio, and storage system performance. The approach uses sliding window techniques to generate the window's hash value based on the file's contents. After that, if the window value fulfills the present criterion, it calculates the chunk cut point according to that value.

To address the drawbacks of the traditional Rabin-based CDC, an improved Rabin-based CDC design is proposed, which increases the accuracy of duplicate detection, reduces chunk size volatility, and lowers computational complexity. This can be achieved through:

1. Adaptive chunking: Implementing an adaptive chunking method that adjusts the size of the chunks based on the content of the data stream. This allows for better deduplication efficiency and reduces the overhead of smaller chunks.
2. Optimized rolling hash function: Proposing an optimized version of the Rabin rolling hash function that minimizes the computational complexity and reduces the chances of false positives in duplicate detection.

The implementation details for the improved Rabin-based CDC technique include:

1. Data pre-processing: Incoming data streams will be pre-processed before applying the improved Rabin-based CDC technique. This can include steps such as data compression, encryption, or normalization to ensure consistency and efficient processing.
2. Sliding window implementation: The sliding window technique will be used to analyze the incoming data stream and generate hash values based on the contents of the file. The improved Rabin rolling hash function will be utilized to reduce computational complexity and increase the accuracy of duplicate detection.
3. Chunk boundary determination and deduplication: When the hash value of the window is equal to a predetermined value, the improved Rabin method will announce a chunk breakpoint. The adaptive chunking mechanism will ensure that chunk sizes are optimized, avoiding tiny and large chunk sizes that may negatively impact deduplication efficiency.

Table 2: Key Properties of various CDC Algorithms – Comparison of overhead

Algorithm	Chunk Size Limits	Efficient Low-Entropy Strings	Computation Overhead	Chunk Variance	Processing Time	Content – Defined
Rabin	Yes	No	High	High	High	Yes
TTTD	Yes	No	High	High	High	Yes
Max P	No	No	High	Low	High	Yes
AE	No	High	Low	Low	Medium	Yes
DPC	Yes	High	Low	Low	Low	Yes

CHUNKING ALGORITHM

Properties of the DPC algorithm

In an iterative algorithm, the asymmetric window located behind the extreme value is delivered by AE, which also breaches the chunk bounds. Keeping them ideally at the chunk border generates the maximum value within each chunk. The methodology is still CDC since the maximum value within the chunks may also reset the chunk boundaries, even if the method can slightly lower the deduplication speed. In contrast to AE, the DPC algorithm dynamically splits the window location depending on prime values. The resulting chunk breakpoint window changes on demand.

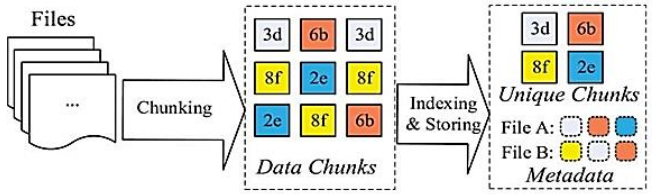


Figure: 2. Chunking Algorithm

The comparative study of the three currently used methods, namely Rabin, MAXP, and AE, is shown in Figure 3 TTTD will not be covered in Fig. 3 since its portrayal in diagrammatic form may be considered quite challenging. As a result, we relied only on three previously developed algorithms. However, the remaining parts of the study compare themselves to four different existing CDC algorithms.

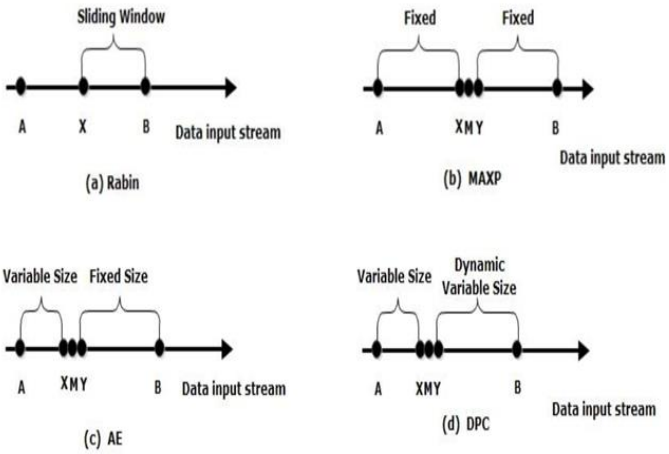


Figure: 3. the difference in the CDC algorithm Rabin, MAXP, AE, and DPC's designs that contribute to the strategic difference

Theoretical Comparison

Rabin is a well-known duplication technique for use with CDC algorithms; nonetheless, it has a very poor chunking throughput and substantial chunk size volatility. The TTTD broke up data into smaller pieces, but it could not pinpoint where data duplication was occurring to account for the larger chunk sizes. In addition, since the processing time has increased, it adds to the overhead associated with indexing. In the end, the chunking AE method was superior to the Rabin regarding the number of low-entropy strings it removed. We suggest using the dynamic prime chunking algorithm to improve the throughput and take the performance to an even higher level.

- Low chunking throughput and time consumption are problems with Rabin.
- The TTTD algorithm adds a minimum and maximum threshold to lessen chunk volatility. The threshold is applied using a backup divisor. For bigger chunks, data deduplication cannot be adequately recognized. Additionally, the longer processing times result in extra expenses for indexing.
- Deduplication efficiency is also much greater in AE. The computational cost is also significantly reduced, and the small chunk variance is raised.

Design Goals

a. Flexibility

When we talk about BenchCloud's Dynamic Prime Chunking utility, we imply that it can be used for various purposes. The fact that various benchmarking jobs have varied targets and purposes drives the need for flexibility, and a generic benchmarking tool should accommodate as many different types of benchmarking tasks as is practically practicable. In order to accomplish this goal, BenchCloud has to have a high level of configurability and extensibility. Being highly adjustable indicates that a user has the ability to make fine adjustments to the parameters of a benchmarking job. These settings can include, for instance, the cloud storage system that he wants to benchmark, the kinds of operations that will be carried out (downloading files, uploading files, etc.), the number of operations that will be carried out, the number of threads that will be used to carry out the operations, and so on.

b. Usability

When we talk about usability, we mean that BenchCloud ought to offer an approachable mode of operation for most of its users. Users should not be required to have a prior understanding of Python programming to use BenchCloud, despite BenchCloud itself being developed in Python. BenchCloud was developed to allow configuration files in order to accomplish this goal. Virtually every aspect of a benchmark's settings may be altered via a configuration file, a simple text document that is not difficult to comprehend or create.

IV. ANALYSIS AND FINDINGS

A. Asymmetric Encryption

A type of cryptographic technique known as asymmetric cryptography calls for using two distinct keys—one secret (or private) and the other public. The two bits of this key pair are numerically associated, although unique. While the secret key is utilized to interpret figure text or lay out a computerized signature, the public key is used to scramble plaintext and approve an advanced mark. Due to the Content Defined Chunking algorithm's reliability and timeliness, we adopted it in our paper.

❖ Content-Defined Chunking Scheme

Content-defined Chunking is another name for it. It might be seen as a DHKE (Diffie-Hellman Key Exchange) protocol development. Unsurprisingly, the discrete logarithm problem's intractability and the Diffie-Hellman (DH) problem's intractability are the foundation for its security. Over the group Z*p, where p is a prime, we consider the content-defined chunking technique. However, it can also be used with other cyclic groups when the DH problem is unsolvable. Content-defined Chunking hypothetically allows techniques including a homomorphic increase on scrambled data. The calculation is the key to creating calculations. After the keys have been generated, the algorithms to encrypt E(x) and decrypt D(x) are now provided in the corresponding sections of Algorithms.

Algorithm 1.0: Content-Defined Chunking

Output: public key $_{pub}$ and private key k_{pr}

1. function KeyGen

2. Choose a large prime p

3. Choose a primitive element $\propto \in Z_p^*$

4. Choose an integer $\alpha \in \{0, \dots p-2\}$

5. $\beta = \propto^{\alpha}$

6. Return$k_{pub} = (p, \propto, \beta)$, $k_{pr} = \alpha$

7. End Function

Input: public key $k_{pub} = (p, \propto, \beta)$ and message m

Output: $ciphertext\ c$

1. Function encrypt (m)

2. Choose $k \in \{2, \dots p-2\}$

3. $x = \propto^k mod\ p$

4. $y = \beta^k * m\ mod\ p$

5. Return $c = (x, y)$

6. End Function

Input: private key $k_{pr} = \alpha$and $ciphertext\ c = (x, y)$

Output: message m

1. function DECRYPT(c)

2. Calculate $m = x^{-\alpha} y\ mod\ p$

3. return m

4. end function

B. Symmtric Encryption

A gathering of cryptographic calculations known as symmetric-key calculations utilizes a similar cryptographic key for the encryption and decoding of plaintext. The two keys might be interchangeable or undergo a straightforward transition. Truly, the keys represent a common mystery that at least two gatherings could use to open a connection to secret data. One of the most outstanding deficiencies of symmetric key encryption, in contrast with public-key encryption is that the two players should approach the mystery key.

C. Upload file

This approach comprises two steps. In the primary stage, the AE calculation scrambles Clair's text. The content-defined chunking calculation is utilized to encode the AE key in the subsequent stage.

Algorithm 2.0: FILE_UPLOAD

1. Encrypt_file (F) {

2. /*algorithm to encrypt file onto cloud storage */

3. /* to transform clair text in file F into Cipher text in file F' */

4. /* Phase 1: Encrypt Clair text with AE Algorithm 6. */

5. for B←1 to numberOfBlock(F) d0

6. {

7. B'=ENC_AE(B,K)

8. }

9. send_to_cloud(F')

10. /* Phase 2: Encrypt AE key with Content Defined Chunking Algorithm */

11. for k←1 to SizeOf(K) do

12. {

13. k'=ENC_Content Defined Chunking (K')

14. }

15. Save_in_server(K')

16. }

D. File Download

This approach has two sections, too; in the primary stage, it utilizes the content defined chunking calculation to disentangle the AE key. The cipher text is decrypted in the second step using the AE key acquired from the server.

Algorithm 2.0: FILE_DOWNLOAD

1. Decrypt_file (F') {

2. /* algorithm to decrypt file downloaded from cloud storage */

3. /* to transfer Cipher text in file F' into Clair text in file F */

4. /* Phase 1: Decrypt AE Key with Content Defined Chunking Algorithm */

5. for k'←1 to SizeOf(K'), do

6. {

7. k=DEC_Content Defined Chunking (k')

8. }

9. return(K)

10. /* Phase 2: Decrypt Cipher text with AE Algorithm */

11. for B'←1 to numberOfBlock(F') do

12. {

13. B=DEC_AE(B',K)

14. }.

15. return(F)

16. }.

E. Results of Dynamic Prime Chunking and Content-Defined Chunking Algorithm Comparison

Figures 4, 5, and 6 show the outcome using a PC with the following specifications (HP Compaq dc 5800): 2 Intel Core 2 Duo E6550 processors running at 2.33 GHz each, along with 3072 MB of RAM.

Figures 4 and 5 show the results of our studies with text files of 64 KB, 100 KB, 128 KB, and 256 KB in size. Because dynamic prime Chunking offers the same level of security on a 1024-bit key size as content-defined Chunking does on a 160 bit, the private key sizes for both algorithms are 1024 bits for dynamic prime Chunkingch and 160 bits for Content Defined Chunking.

Figure 4 compares the encryption times, measured in seconds, for dynamic prime Chunking and content-defined Chunking. Dynamic prime Chunking performed better than content-defined Chunking, but the encryption rates of the two techniques are similar. Content-defined Chunking outperformed dynamic prime Chunking in decryption, as seen in figure 4; however, the two algorithms are not comparable due to the vast time gap between dynamic prime Chunking and content-defined Chunking.

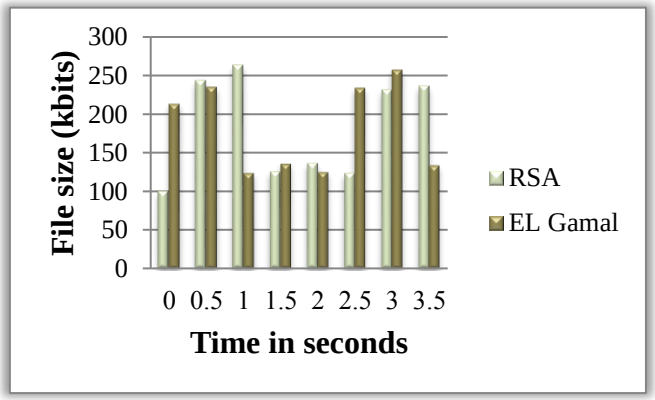


Figure: 4. Comparison of Dynamic Prime Chunking

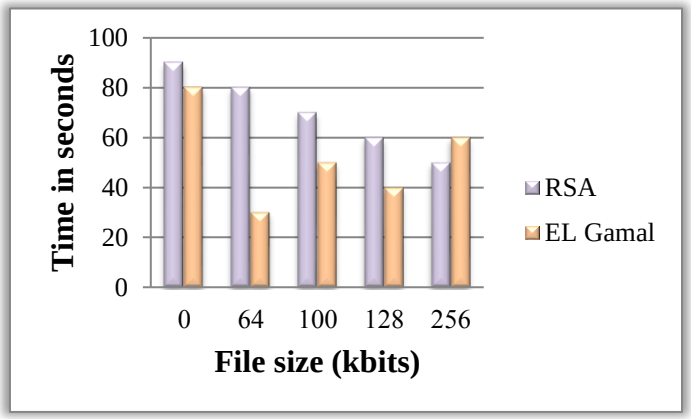


Figure: 5. Decryption times for Dynamic Prime Chunking and Content Defined Chunking are compared

F. Implementation results

The implementation of the results in this part emphasizes how long it takes to upload and download files of various sizes. More time is spent downloading than uploading. The incorporation of key recuperation time on the server makes sense of this.

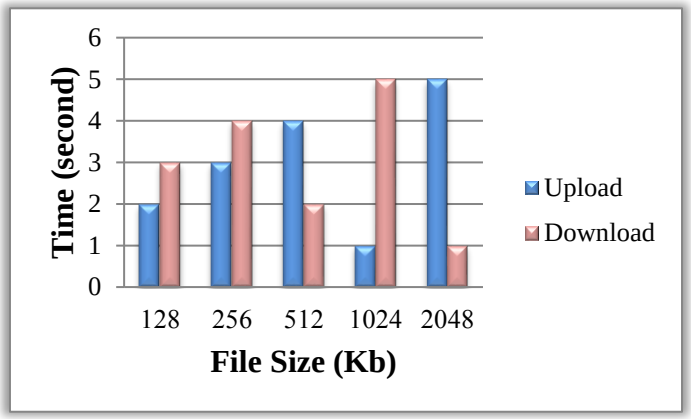


Figure: 6. Execution Time for File Algorithms for Upload and Download

G. Proposed Algorithm Benefits

The followings are some benefits and strong points of our algorithm:

- Data is encoded from the source machine to the objective machine; the public cloud doesn't contain the unscrambling key.
- The AES algorithm, which is one of the safest and fastest symmetric algorithms, is employed. It has not yet been broken. This demonstrates that our technique is fast in both transfer and download.
- The ability to often swap the symmetric key to increase security.
- The content-defined chunking algorithm is used to encrypt the AE key used for data encryption. It is a reliable, probabilistic method that has never failed.

- Data decryption necessitates two-factor authentication; the user must be granted access to the organization's server and cloud storage.

CONCLUSION

Despite the many benefits of cloud storage, numerous security issues still need to be resolved. If we can dispense with or ace this security shortcoming, cloud storage will be the future for both enormous and independent ventures. The various cloud computing vulnerabilities were discussed in this article, and we also suggested how to increase the security of data storage by implementing our algorithm. Access to the data is restricted to the authorized user. The algorithm's primary objective is to secure cloud-stored data. The creators utilized a symmetrical calculation for this. They further developed the calculation utilizing various methods with the goal that it may be utilized effectively to scramble data stored in the cloud. Since this calculation will likewise be utilized to scramble private data, symmetric algorithms were chosen by the authors because they are frequently used to store private data globally. The creators' calculation guarantees data validity, secrecy, and uprightness for data put away in the cloud.

RECOMMENDATION

Key recommendations for the upcoming work are listed below:

Future studies must concentrate on moving a small portion of the network with public data to the cloud for testing purposes in order to advance the field's understanding. This research may be expanded to standardize cloud computing, which becomes a component of every company, to obtain feedback at the domain and cloud service provider levels and identify the attack path at a specific time with an attack alert. Further study should examine how encrypted indexes can reduce the time needed to execute user requests.

REFERENCES

[1] Aditham, S., &Ranganathan, N. (2017). A System Architecture for the Detection of Insider Attacks in Big Data Systems. IEEE Transactions on Dependable and Secure Computing, 1–1. doi:10.1109/tdsc.2017.2768533

[2] B. Waters, D. Balfanz, G. Durfee, and D. Smetters, "Building an encrypted and searchable audit log," in ISOC Network and Distributed System Security Symposium (NDSS 2004). Citeseer, 2004.

[3] BhavaniBai, B. (2014) Ensuring Security at Data Level in Cloud Using Multi-Cloud Architecture. International Journal of Science and Technology.

[4] Daemen, J., & Rijmen, V. (2013). The design of Rijndael: AES-the advanced encryption standard. Springer Science & Business Media.

[5] H. Takabi, J. B. D. Joshi and G. Ahn, Security and privacy challenges in cloud computing environments, Security & Privacy, IEEE, Vol. 8, no. 6, pp. 24–31,(2010).

[6] Jose, N. and Kamani, C. (2013) A Data Security Model Enhancement in Cloud Environment. Journal of Computer Science and Engineering, 10, 1-6.

[7] L. Arockiam, S. Monikandan « Data Security and Privacy in Cloud Storage using Hybrid Symmetric Encryption Algorithm » International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 8, August 2013

[8] L. Wei, H. Zhu, Z. Cao, X. Dong, W. Jia, Y. Chen, A.V. Vasilakos, Security and privacy for storage and computation in cloud computing, Inform. Sci. 258 (2014) 371–386.

[9] Mounica, D. and Rani, R. (2013) Optimized Multi-Clouds Using Shamir Shares. International Journal for Computer Science & Technology Development, 1, 83-87.

[10] News Briefs, Amazon's Massive Cloud Hosting Site Crashes, Journal of IEEE Computer, Vol. 44, pp. 18-20,(2011).

[11] P. Arora, R. C. Wadhawan, and E. S. P. Ahuja, « Cloud Computing Security Issues in Infrastructure as a Service,» Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 2, no 1, 2012.

[12] Papri Ghosh, Vishal Thakor, Dr. Pravin Bhathawala (2017). "Data Security and Privacy in Cloud Computing Using Different Encryption Algorithms" International Journal of Advanced Research in Computer Science and Software Engineering 7(5), May, pp. 469-471.

[13] R.D. Dhungana, A. Mohammad, A. Sharma, I. Schoen, Identity management framework for cloud networking infrastructure, in IEEE International Conference on Innovations in Information Technology (IIT), 2013, pp. 13–17.

[14] Sathana, V. and Shanthini, J. (2014) Automated Security Providence for Dynamic Group in Cloud. International Journal of Innovative Research in CE, 2.

[15] Shalu Mall, Sushil Kumar Saroj (2018), A New Security Framework for Cloud Data, Procedia Computer Science, 143, 765-775

[16] M. Ellappan and S. Abirami, "Dynamic Prime Chunking Algorithm for Data Deduplication in Cloud Storage," KSII Transactions on Internet and Information Systems (TIIS), vol. 15, no. 4, pp. 1342-1359, 2021, http://doi.org/10.3837/tiis.2021.04.009.

[17] P. Anitha, R. Dhanushram, D. H. Sudhan, and T. Indhresh, "Security Aware High Scalable paradigm for Data Deduplication in Big Data cloud computing Environments," in Journal of Physics: Conference

[18] Z. Xu and W. Zhang, "QuickCDC: A Quick Content Defined Chunking Algorithm Based on Jumping and Dynamically Adjusting Mask Bits," in 2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), 2021, pp. 288-299: IEEE, DOI 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00049.

[19] N. Kumar and S. Jain, "Efficient data deduplication for big data storage systems," in Progress in Advanced Computing and Intelligent Engineering: Springer, 2019, pp. 351-371, https://doi.org/10.1007/978-981-13-0224-4_32.

[20] Xia, W., Zhou, Y., Jiang, H., Feng, D., Hua, Y., Hu, Y., Liu, Q., & Zhang, Y. (2016). FastCDC: a Fast and Efficient Content-Defined Chunking Approach for Data Deduplication. USENIX Annual Technical Conference.

[21] Puri, G.S., Tiwary, R. and Shukla, S. (2019). *A Review on Cloud Computing*. [online] IEEE Xplore. doi:https://doi.org/10.1109/CONFLUENCE.2019.8776907.

[22] Attaran, M. and Woods, J. (2018). Cloud computing technology: improving small business performance using the Internet. *Journal of Small Business & Entrepreneurship*, [online] 31(6), pp.495–519. doi:https://doi.org/10.1080/08276331.2018.1466850.

[23] Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H. and Zhao, W. (2017). A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. *IEEE Internet of Things Journal*, 4(5), pp.1125–1142. doi:https://doi.org/10.1109/jiot.2017.2683200.

[24] Qiu, H., Ji, T., Zhao, S., Chen, X., Qi, J., Cui, H. and Wang, S. (2022). A Geography-Based P2P Overlay Network for Fast and Robust Blockchain Systems. *IEEE Transactions on Services Computing*, [online] pp.1–14. doi:https://doi.org/10.1109/TSC.2022.3189667.

[25] Ahmed, M. and Sarkar, N.I. (2020). *Privacy in Cloud-Based Computing*. [online] Social, Legal, and Ethical Implications of IoT, Cloud, and Edge Computing Technologies. Available at: https://www.igi-global.com/chapter/privacy-in-cloud-based-computing/256267 [Accessed 31 Mar. 2023].

[26] Dong, J., Wang, H., Li, Y. and Cheng, S. (2014). Virtual machine placement optimizing to improve network performance in cloud data centers. *The Journal of China Universities of Posts and Telecommunications*, [online] 21(3), pp.62–70. doi:https://doi.org/10.1016/S1005-8885(14)60302-2.

[27] Tancock, D., Pearson, S. and Charlesworth, A. (2012). A Privacy Impact Assessment Tool for Cloud Computing. Computer Communications and Networks, pp.73–123. doi:https://doi.org/10.1007/978-1-4471-4189-1_3.

[28] Sultan, N. (2013). Knowledge management in the age of cloud computing and Web 2.0: Experiencing the power of disruptive innovations. *International Journal of Information Management*, 33(1), pp.160–165. doi:https://doi.org/10.1016/j.ijinfomgt.2012.08.006.

[29] Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J. and Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6), pp.599–616. doi:https://doi.org/10.1016/j.future.2008.12.001.