JSR Publications

Available Online

J. Sci. Res. **14** (1), 153-165 (2022)

JOURNAL OF SCIENTIFIC RESEARCH

www.banglajol.info/index.php/JSR

Data Security and Deduplication Framework for Securing and Deduplicating Users' Data in Public and Private Cloud Environment

K. Balaji*, S. S. Manikandasaran

PG and Research Department of Computer Science, Adaikalamatha College, Vallam, Thanjavur, Tamil Nadu, India. Affiliated to Bharathidasan University

Received 17 June 2021, accepted in final revised form 14 October 2021

Abstract

Maintaining the security of data stored in the public or private cloud is a more tedious task. The cloud is the only arrangement for storing enormous amounts of data, but there is a possibility of storing the same data more than once. The traditional security system generates different unreadable data for the same readable content of a file. Therefore, it is necessary to address data security of the cloud and duplication in cloud storage. This paper concentrates on developing a data security and deduplication framework with different security techniques and mechanisms to address the said difficulties in the cloud. The framework proposed in this paper focuses on reducing security vulnerability as well as data duplication. The paper describes the components used in the frameworks. The main research contribution of the framework is having enhanced the convergent encryption technique, key generation techniques, and deduplication mechanism for maintaining a single copy of data in the cloud. The proposed framework's efficiency is measured by implementing the work by developing a cloud-based application that coded for all the procedures of the proposed framework and tested in the cloud environment.

Keywords: Data security; Deduplication; Convergent encryption; Public cloud; Private cloud.

doi: http://dx.doi.org/10.3329/jsr.v14i1.54063 J. Sci. Res. **14** (1), 153-165 (2022)

1. Introduction

Cloud is the latest technology that provides computing resources through the Internet. It is Internet-based computing. Internet is the basic requirement to use cloud services. This technology is not entirely new. Evolutionary technology forms pervasive computing, parallel computing, grid computing, utility computing, etc. It has enormous computing power to provide huge virtual storage. It provisions the computing resources in three ways: software, platform, and infrastructure services. Cloud is a place with a huge amount of computing infrastructure built by lots of high-power servers and network connections. It is running in a 24×7 manner. It mainly helps small and medium scale enterprises to satisfy their computing needs. Cloud is more reliable for storing data, and it is more

* *Corresponding author*: balajjee.mecse@gmail.com

efficient to deliver the data at the request [1]. Cloud computing has its essential characteristics: on-demand self-service, broad network access, Multitenancy, Elasticity, and Metered service. It provides an unlimited computing service to users based on their requests. Apart from the many benefits of the cloud, it is suffered by some darkest parts of the cloud. Management of cloud storage is a critical task in the cloud, and more specifically, securing data in cloud storage is a noticeable challenge in the cloud [2].

Cloud ensures reliability because it stores multiple copies of a single piece of data. Therefore, the storage contains many duplicate data. Traditional cryptosystems lead to the store of duplicate data in the cloud. Duplicate data in cloud storage makes administration overhead in the cloud. Storing Duplicate data wastes the bandwidth, and it leads to performance issues. Therefore, secure deduplication of the cloud is a must to maintain cloud storage efficiently. Data deduplication methods were put in place to support overall compression to approximate granularity [3].

Previous methodologies are very ineffective in the relative block of identity and are not versatile. Information deduplication policies may be operational at the file or sub-file level and the block level [4]. It encloses information using pieces of variable or fixed size. Cryptographic hashing measures generate hash estimates for these blocks, and copies are identified by coordinating hash standards [5]. The deduplication technology was analyzed to determine performance, efficiency, challenges, and cost. However, the primary system's deduplication techniques have not yet been as effective as the relief deduplication technology [6]. Because the deduplication was carried out in the primary system 68 %, but in the backup system 83 %, the backup deduplication method has significantly increased data storage efficiency. Higher concentration must reduce deduplication in the backup storage system [7]. Data deduplication techniques are categorized at both block and file levels. At the file level, the files are compared depending on their originality [8]. According to fixed or variable length, data burst into small pieces at the block level and find the block's original copy. Deduplication is performed on either client or server-side. Client-side deduplication safeguards storage waste and records network bandwidth. Server-side deduplication saves storage space and reduces overall user-side deduplication system operating costs. Deduplication is supported by convergent encryption. Convergent encryption uses a convergent encryption key for encrypting and decrypting the data before storing it in the cloud storage. This convergent encryption key is generated from the user's data to be stored in the cloud storage. The convergent key ensures that whether the data is already stored in the cloud or not [9]. This paper designs a framework for secured deduplication to ensure efficient data storage in the cloud.

2. Related Work

Cloud storage is more efficient for storing a huge amount of data. However, a security vulnerability in cloud storage is a huge hurdle for users to store the data in the cloud. To maintain security and deduplicated cloud storage, different researchers undertake many

types of research. This is section discusses some of the selected research work already done by the same field. Lashkari *et al.* [10] proposed a framework with data ownership challenge-based functionality for deduplication to manage textual encrypted data files. This approach verifies file data ownership and verifies duplicate file content with a security challenge and big data support. This framework includes the following components: Users: -First, users have to create a login and password for individual data privacy. Upload: -After a registered user logs in, the user uploads the data to the cloud on which the operations are carried out. Deduplication check: When downloading the file, the file is checked for duplication. Generation of keys: The generation of keys is carried out according to the content present in the file. Server: Afterwards, the encrypted file is stored in the cloud. The file is accessible through the cloud. File sharing and access: If a user wishes to access another user's file, the file key is shared with the particular user. Then only the user has access to the corresponding file. Admin: Admin keeps track of all actions of all users.

Kumar [11] proposed a framework composed of eight components: client, file uploader, proof of ownership, cloud service provider, cloud database, file uploader, and user. The work's main objective is to design a secure cloud data system that benefits customer data (cloud data storage user) and server data (cloud storage provider). For user benefit, and improved cloud data protection mechanism for the information stored in the cloud is provided by encrypting the data before uploading the file to the depot server. Simultaneously, cloud storage capacity is optimized by avoiding redundant storage of the same file on the server. Encrypting the same file with two different encryption keys generates two different encrypted files, which affects the cloud service provider's deduplication concept. Thus, an efficient mechanism generates the hash value from the text file's content and then uses it to encrypt the data. The hash value with a single file identifier is stored in the database to avoid data attack leaks. The server creates a link to the file without storing redundant data to perform deduplication. The proposed work also effectively uses the customer's bandwidth because, when downloading a file redundantly, only the hash value is passed to the server and checked.

Devarajan *et al*. [12] proposed a Storage Optimization System (SOS) framework to decrease the storage allocation using Deduplication Compression (DDC) algorithm to improve the storage and utilization of bandwidth in the SOS through the elimination of duplicate files and monitor the IaaS storage utilization. The different metadata standards help identify the respective file objects, and the file data elements bound to the corresponding block can be grouped into segmented bins. The DDC algorithm is associated with a linked list structure to access the file on different parameters to classify future file access models related to cloud storage. The ranking was determined based on user access frequency to predict future usage based on file access models. The SOS framework has a dashboard that can help the user to decide how the file will work. The customized Storage Optimization System utilizes the deduplication technique to pre-process file/data storage in IaaS storage in the cloud. The following four processes are

used as part of the deduplication process.1) Segmentation, 2) deduplication verification, 3) metadata management, 4) data storage to identify a redundant copy of the file.

Manikandasaran *et al*. [13] proposed a technique named MONcrypt is based on the obfuscation technique. The authors have discussed the security issues in the cloud and mention the importance of security in cloud storage. They use the obfuscation technique in their proposed technique. Obfuscation is a process of masquerading the original text into irrelevant text without using a key, contrary to encryption. Normally, traditional obfuscation is not using and but MONcrypt uses a key for obfuscating the data. This new obscuring technique is used to ensure the confidentiality of externalized data in cloud storage. The authors tried to implement the work in a real-time cloud environment. They use hacking tools to analyst the security of the data. The proposed technique shows improved performance and safety concerning existing techniques.

George *et al*. [14] proposed GLEnc encryption. It aims to encrypt the data before it enters into cloud storage. The method focuses on protecting data in the Public Cloud Environment. The proposed model enables a simple plain text to undergo different types of splitting and dividing techniques. Three keys are employed at different stages to get highly encrypted data. First, the plain text values are counted and converted to ASCII decimal values and then binary. Then, these values undergo splitting and dividing with bit conversion. Symmetric encryption uses the same keys in the encryption and decryption of the data. Therefore, GLEnc produces different ciphertexts for the identical plain text that happened more than once in the plain text content. Thus, ensuring the proposed GLEnc algorithm high security of public cloud environment.

Uma *et al*. [15] delivered a technique used to generate a convergent key from the user data, which is to be uploaded to the cloud, and the user's data are encrypted with this convergent key. Existing convergent encryption techniques are vulnerable to different attacks and not suitable to cloud storage. The proposed work is an enhanced symmetric convergent encryption technique to maintain secure deduplication with convergent encryption in the cloud. The paper introduces a new approach to getting the cloud's key and encryption technique as a service. Encryption is done on the users' side. The proposed technique is analyzed for a security vulnerability, and it ensures the security and efficient deduplication by the procedure used in the overall data deduplication process

Pusukuri *et al*. [16] represented the importance of the data compression technique in data deduplication. In this technique repeating data of duplicate copies are eliminated. To support deduplication, protect the confidentiality of sensitive data. The paper has been proposed a convergent encryption technique to encrypt the data. For better protection of data security, identify the problem of authorized data deduplication in the first attempt. There have different traditional deduplication systems. From those systems, different privileges of users are further considered in duplicate checks besides the data itself. In hybrid cloud architecture, present several new deduplication constructions for supporting authorized duplicate checks. Based on the definitions, the paper proposed a security model for security analysis. A proof of concept is implementing a prototype of the authorized duplicate check and using our prototype. A testbed experiment is conducted.

Agarwala *et al*. [17] proposed a secure data deduplication protocol, DICE (Dual Integrity Convergent Encryption), in which it is focused on i) to prevent the duplicate-faking and erasure attacks, and ii) to provide integrity check at both client and server ends. Generated tag is uploaded; the server performs the integrity check of the received tag. The integrity check is performed when the client downloads the tag to retrieve the ciphertext. As a result, reduce the bandwidth consumption by sending only the tag instead of the long ciphertext while at the same time achieving deduplication. The key generation and management processes are not clearly described in this paper.

Periasamy *et al.* [18] presented an enhanced secure content deduplication identification and prevention (ESCDIP) approach to improve the file-level and content-level deduplication discovery of coded data with consistency in the cloud environment. Every cloud user's files comprise an independent master key for encryption using the ESCDIP technique and outsourcing them into the cloud. It shrinks the overheads that are connected with the collaborative duplication detection and query processes. The method identifies the unique data chunking to store in the cloud. The cloud user sets the size of the data chunk. Chunk is also denoted as segments. Each segment is assigned with an identifier. The method identifies the duplicated data by comparing the data segment identification, where only a copy of every repeated part will be stored. At any cost, duplicated segments cannot be saved, and pointers are designed for them. Thus, the technique improves storage efficiency and reduces backup costs. Based on the result, this method deduced data uploading and downloading time and minimize communication costs compared with other methods.

3. Problem Definition

The cloud allows duplicate data to store in the cloud storage. As a result, it creates unwanted storage allocation in the cloud. They are forced to use the platform and infrastructure provided by the same CSP for all services. It requires the use of interfaces provided by the CSP. Users' data have to be in a fixed format specified by the CSP. Users' data sent to the cloud are controlled and monitored by CSPs. CSPs know where the users' confidential data are located. CSPs have the privilege to access and collect the users' confidential data. Cloud storage is allocated for storing the same data multiple times. As a result, it creates difficulties in storage management. The traditional cryptosystem is not suitable for maintaining non-duplicate data in the cloud storage environment. The deduplication mechanism is vulnerable to brute force attacks, dictionary attacks, and poison attacks, including duplicate faking attacks and erasure attacks [19]. The convergent encryption key should be the same for all users who have the same data, but sharing this key with all the users of the data owner is vulnerable to a security attack.

4. Methodology

The paper proposes a data security and data deduplication framework to secure and duplicate users' data in public and private cloud storage. The objective can be achieved by

proposing different convergent encryption techniques for public and private cloud storage. Furthermore, the proposed framework comprises a convergent key generation technique to generate and maintain the keys. Finally, the data are verified by the Data Deduplication mechanism, which ensures the duplicate copy of the data. Fig. 1 illustrates the block diagram for the proposed safety and data deduplication framework.

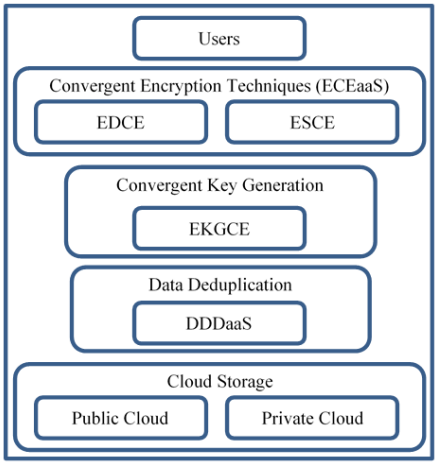

Fig. 1. Block diagram of the proposed framework.

The proposed framework consists of different cloud services for cryptography service, key generation, and maintenance service, Data deduplication service for the public, private cloud storage service. Cryptography is provided as a service in the name of Enhanced Convergent Encryption as a Service (ECEaaS). The key generation is provided by the cloud service called Key Manager as a service (KMaaS). KMaaS is a key generation method. It generates a key from the users' data. It is also called Convergent Encryption (CE) key. The data uploaded to the clouds are verified for duplication to avoid data redundancy in the cloud storage. Data DeDuplication as a Service (DDDaaS) is proposed in the framework for duplication verification on public and private cloud data. The data may be stored in a public or private cloud based on the users' interests.

5. Proposed DSDD Framework

The proposed Data Security and Deduplication Framework (DSDDF) has a cloud service that provides security service using symmetric cryptographic encryption, known as the Enhanced Convergent Encryption (ECEaaS). ECEaaS consists of two cryptographic symmetric encryption techniques: Enhanced Data Convergent Encryption (EDCE) and Enhanced Symmetric Convergent Encryption (ESCE); the techniques are proposed for the public and private cloud environment. Generally, there are four types of cloud deployment models: public, private, hybrid, and community cloud environment. Apart from that, this research work focuses on providing security to public and private cloud data.

Cryptography techniques address security. For cloud storage, a symmetric cryptosystem is more suitable and efficient to keep data safe. However, asymmetric is not suggested for encrypting a huge amount of data stored in the cloud.

Different clouds and cloud services are designed in the proposed framework. Mainly, security services provided by the ECEaaS and KMaaS are the scope of this work. Along with ECEaaS and KMaaS, another cloud service is included in the framework for verifying data deduplication called DDDaaS. The DDDaaS maintains all the details about the data stored in the cloud. It has database details about the file, convergent key, tag, etc. Using the convergent key and the tag, DDDaaS verifies the deduplicated data. KMaaS is a key providing service to the encryption techniques in the ECEaaS. The encryption techniques used in the framework are convergent. The key for the encryption is also a convergent encryption key. The key is generated for the user based on their wish with the help of KMaaS. Based on the encryption technique chosen by the user, keys are generated from data uploaded to the cloud. The framework separates the cloud services, which means security, key, deduplication, and storage services. Independent services providers provide all these services. Different independent cloud service providers provide these services. Hence, the storage providers do not know which security techniques and the key to hide the data. Likewise, the key service provider does not know in which cloud the data are stored. This scenario avoids the security risk of the data stored in cloud storage.

The framework considers the storage of data in the public cloud and private cloud. For each cloud data, there is a different convergent encryption technique proposed in ECEaaS. First, users have to decide which cloud they will upload the data to; they must choose a symmetric convergent encryption technique to encrypt the data. The following Fig. 2 shows the diagrammatic view of the proposed security framework for cloud environments.

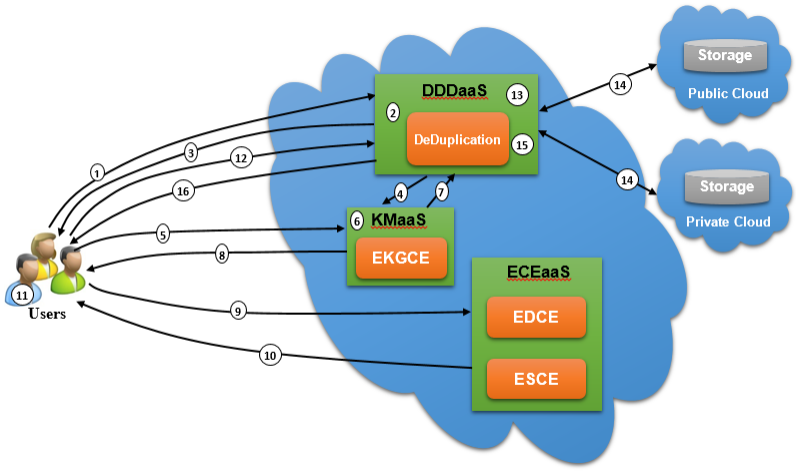

Fig. 2. Data security and data deduplication framework.

The procedural steps followed in the entire proposed framework are given below.

Steps in the process flow of proposed framework

1. Users send data uploading requests to the proposed framework through applications running on the client-side. The initial request is given to DDDaaS.
2. The DDDaaS in the framework generates a token for the data and checks whether the token is already presented in its database.
3. If it is not already presented in the database of DDDaaS, then the token sends to the user. Further, the following steps are carryout.
4. DDDaaS forwards the generated token to KMaaS.
5. User requests a convergent key from the KMaaS by submitting the token of the file given by DDDaaS.
6. KMaaS generates the key.
7. The key is forwarded to DDDaaS.
8. The key is forwarded to the user.
9. User requests convergent encryption for encryption.
10. The ECEaaS provisioned the required encryption for users.
11. User encrypts the data using the key and selected convergent encryption.
12. User uploads the encrypted data to the DDDaaS for storing it in the cloud.
13. DDDaaS generates a tag for the encrypted data, then checks whether the tag is already presented in its database. The encrypted data is not stored when the tag is already presented in the database.
14. The encrypted data are stored in the public or private cloud storage if the tag is not presented in the DDDaaS database.
15. If the token generated initially is already presented in the DDDaaS database. Further, the following steps are carryout.
16. The DDDaaS does not allow the user to generate the key and encrypts the data. Instead, it verifies the user's authenticity and allows the current user access to the already stored file in the cloud storage.

Each time a data uploads to cloud storage, it is encrypted using convergent encryption. For encrypting the data, a key is generated from the data being uploaded to the cloud. Consider the following cases to understand the framework further.

Case 1: A new data file uploads to the cloud storage. A token is generated using DDDaaS for data file content. Now, the generated token is verified with the DDDaaS database. Whether the token does not exist in the DDDaaS database, then DDDaaS allows the user to generate the convergent key. The key is the digested value of the data being uploaded to the cloud. Using this convergent key, the data is encrypted. After the encryption, the encrypted data is submitted to the DDDaaS. The DDDaaS generates a tag for encrypted data and checks whether the tag already exists in its database. If it does not exist in the database, then the encrypted data is stored in the cloud. Suppose the tag exists

in the DDDaaS database already, then it shows that the encrypted data is previously stored in the cloud storage.

Case 2: A data file uploads to the cloud storage. A token is generated using DDDaaS for the data file content. Now, the generated token is verified with the DDDaaS database. Whether the token exists in the DDDaaS database shows that the specific data file is already stored in the cloud storage by another user. Now DDDaaS verifies the user authenticity of the user to the data file and allocates the file access link to the new user. However, the data is not uploaded to the cloud for a second time.

The framework strongly protects the data from unauthorized access, and it also avoids multiple copies of data stored in the cloud. Furthermore, the framework ensures that the same data is not stored in the public and private cloud. The encryption techniques proposed for public and private clouds use the same keys for encryption and decryption. The procedure below outlines the steps for the proposed key generation technique.

Steps to be involved in the generation of CE key

1. Users' data are input PT.
2. Find the length of the PT.
3. The characters in the PT are converted into decimal code.
4. Divide the PT into 16 characters' lengths of the blocks.
5. Add each block decimal with the following 16 block decimal values.
6. Find the modules by dividing each decimal value by 256, note the remainder value.
7. The derived 16 reminder values are converted into binaries 0's and 1's.
8. Finds the one's complement of the 128 bits binary.
9. Divide the 128 bits into 8 bits.
10. Count the number of 1's in each 8-bit block.
11. Rotate left to right each 8bits at the number of times 1's in every 8 bits.
12. Convert the 8bits into the ASCII decimal code.
13. Convert the decimal into the character code.
14. The converted character code is the 128-bit CE key.

The key generation is invoked after the user-chosen the convergent encryption. Then, users directly contact the key generation services to get the key. Key received from the KMaaS is not known to the ECEaaS. Thus, ECEaaS comprises two convergent encryptions for public and private cloud storage. Then, based on the user's wish, the corresponding procedure is invoked to encrypt the data. For example, the following procedure involved convergent encryption in ECEaaS.

5.1. *Procedure to be followed in the proposed EDCE*

This procedure is used for encrypting the data which are going to store in public cloud storage. The users' data are the plaintext to upload. The plaintext is converted into binaries. Divide the total length of plaintext bits into 128-bit blocks. The EDCE cipher takes the 128-bit binaries as input for encryption. If the plaintext has more than one 128

bits block, every 128 bits are encrypted using the same subkeys. EDCE cipher generates a round number key, which denotes the number of rounds. The encryption procedure of the EDCE cipher is as follows,

1. The 128 bits are divided into 64 bits blocks called LD1 and RD1
2. Find the Left circular shift for LD1 for the number of times bit 1 is in the 64 bits.
3. Find the Right circular shift for RD1 for the number of times the bit 1 is in the 64 bits
4. Find XOR with subkey LK1 with LD1 and RK1 with RD1
5. Find the XOR for RD1= LD1 XOR RD1
6. Merge the two 64 bits into a single 128 bits
7. The find reverse the 128 bits
8. Find one's complement. Now a single round is completed

5.2. *Procedure to be followed in the proposed ESCE*

This procedure is used for encrypting the data which are going to store in private cloud storage.

1. User's data are transformed into equivalent binary.
2. Binaries are divided into 128bit blocks.
3. A Key K1 is generated from the cloud.
4. Alternatively, Insert the bits of key K1 into each block of 128bit
5. Find 1's complements on each block
6. Divide the 256bit blocks into 128bit blocks and use the same key K1 to find XOR with each 128 bits block.
7. Split each 128bit block into a 64bit block
8. Find the 1's complement on each 64bit block
9. Join the pair of two 64bits blocks into one 128bit block
10. Find the reverse of every 128bit block and Merge all the blocks into a single block
11. Split block into two blocks and Merge all blocks into a single block
12. Extract alternate 8bits from the 512 bits, which means extract odd 8 bits and even 8bits separately
13. The extracted odd 8 bits of 256 bits are converted into decimal and ASCII character code. It is a key K2 and kept safe.
14. The extracted even 8 bits of 256 bits are converted into decimal and ASCII character code. It is the encrypted data for plain text.

All the proposed procedures discussed in this paper are coded and developed as an application and hosted in the real-time cloud environment. The effectiveness of the proposed procedures is calculated based on the computation time caused by the procedures. The following section describes the implementation environment used for this research work.

6. Implementation Setup and Results

The proposed research work is coded and developed as a cloud-based service. The developed application is deployed in the MyASP.NET cloud server. MyASP.NET is a cloud-based platform providing service that enables the users to deploy their application to get a cloud-based experience. The cloud-based application is developed and hosted on the cloud server. The proposed application is developed in C#.NET using visual studio 2012. The hosted cloud application is served as a cloud service consumed by all procedures proposed in the research work. Fig. 3 shows the diagrammatical representation of the implemented setup created for research work.

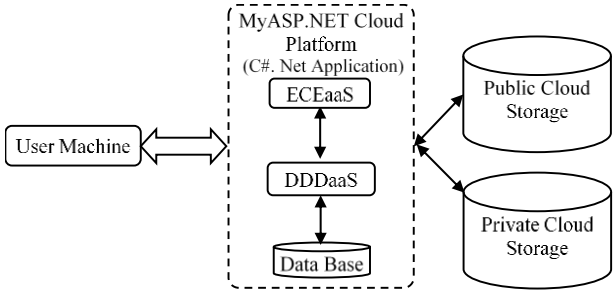

Fig. 3. DSDDF implementation setup.

Convergent encryption, convergent encryption key generation, and tag generation all these procedures are all running in the developed cloud-based application. For storing data, the MyASP.NET cloud server is used in the implementation. The implementation shows that if the same data file is uploaded for the second time, it gives an error message that the file is already available in the storage and please try another file. At the same time, the data is not uploaded to cloud storage. Thus, it proves that the proposed framework is not allowed to upload duplicate data. Hence, it enables the effective management of data in cloud storage.

The efficiency of the proposed approach is measured by considering the computational time caused by the proposed and existing methods. It can be calculated by the time taken by the data security and deduplication techniques for uploading data to the cloud. Calculation includes the processing of all steps describes in the framework. A comparison of computational time considers the different sizes of data. The computation time includes the running of all the procedures when uploading the data. The paper provides a comparison based on two cases discussed in the previous section.

Table 1 shows the comparison of proposed and existing approaches concerning computational time based on *Case 1*. In this case, a new data file is uploaded to the cloud storage. The computation time is calculated based on token generation, encryption, key generation, and tag generation procedures. The result illustrates that the proposed method has taken a minimum computational time to upload a new data file compared to other existing plans.

Table 1. Computational time caused by the Proposed and Existing Secured Deduplication concerning Case 1.

Size (KB)	Dekey [20]	Dupless [21]	DSDDF
	Milliseconds		
100	299	280	163
200	592	556	324
300	886	831	483
400	1179	1108	645
500	1473	1284	804

Table 2 shows the comparison of proposed and existing approaches concerning computational time based on *Case 2*. In this case, a user trying to upload a data file is already stored in cloud storage by another user. When the file already exists in the storage, it is not necessary to upload it again. Therefore, the computation time for this case is calculated only based on the running time of the token generation. If the generated is matched with the DDDaaS database, then no other procedures are invoked. It illustrates that the proposed method has taken a minimum computational time to upload a data file already stored in the cloud compared to other plans.

Table 2. Computational time caused by the proposed and existing secured deduplication concerning case 2.

Size (KB)	Dekey	Dupless	DSDDF
	Milliseconds		
100	47	28	12
200	95	55	25
300	140	85	37
400	186	112	46
500	235	139	55

The result from the implementation of the proposed research shows that the efficiency of the proposed framework. Further, the work will be tested for security analysis soon and also the deduplication system will be implemented in the mobile cloud environment.

7. Conclusion

A security and deduplication framework is proposed with different components for security, key generation, and deduplication. The main focus of the research is to propose enhanced techniques and mechanisms for security and deduplication, respectively. The security techniques are based on symmetric convergent encryption, which helps to ensure security and deduplication. Deduplication is the process of verifying data duplication and enables cloud storage with a single copy of data. The framework does not support duplicate data in the cloud. It avoids the workload of key maintenance users. Hence, users are not required to generate and maintain the framework's elements, unlike existing deduplication architectures. Converged encryption, key generation, deduplication, and storage mechanism are delivered as a cloud service. As a result, save time, bandwidth and

keep management headaches to a minimum. The framework saves the storage from the redundancy allocation. The proposed framework minimizes the computation time. It enables cloud providers to maintain their storage environment effectively.

References

1. B. Alouffi, M. Hasnain, A. Alharbi, W. Alosaimi, H. Alyami, and M. Ayaz, IEEE Acc. **9**, 57792 (2021). https://doi.org/10.1109/ACCESS.2021.3073203
2. P. Silva, E. Monteiro, and P. Simões, IEEE Acc. **9**, 10473 (2021). https://doi.org/10.1109/ACCESS.2021.3049599
3. D. Zhang, J. Le, N. Mu, J. Wu, and X. Liao, IEEE Trans. Cloud Comput. (2021). https://doi.org/10.1109/TCC.2021.3081702
4. Pronika and S. S. Tyagi, Int. J. Inno. Tech. Expl. Eng. (IJITEE) **9**, 364 (2019). https://doi.org/10.35940/ijitee.B1027.1292S19
5. A. S. Jenitha, V. S. J. Prakash, Int. J. Recent Tech. Eng. (IJRTE) **8**, 4084 (2019). https://doi.org/10.35940/ijrte.C5453.098319
6. L. Suresh and M. A. Bharathi, Analysis of Block-Level Data Deduplication on Cloud Storage, in Ambient Communications and Computer Systems, Advances in Intelligent Systems and Computing, ed. Y. C. Hu et al., (Springer, Singapore, 2019) **904**. https://doi.org/10.1007/978-981-13-5934-7_36
7. A. Vijayakumar and D. A. N. Jebaseeli, Int. J. Grid Dist. Comp. **13**, 1 (2020).
8. C. Yu, C. Chen, and H. Chao, IEEE Network **29**, 51 (2015). https://doi.org/10.1109/MNET.2015.7064903
9. Y. Zhang, C. Xu, N. Cheng, and X. S. Shen, IEEE Trans. Depend. Sec. Comp. https://doi.org/10.1109/TDSC.2021.3074146
10. M. Lashkari, A. Suntnure, Siddheshwar, S. Kathale, and H. Wankhede, Int. J. Res. Eng. Sci. Mgt. **2**, 5 (2019).
11. S. Muthurajkumar, Int. J. Eng. Adv. Tech. (IJEAT) **9**, 1 (2019). https://doi.org/10.9790/0661-1904052529
12. A. A. Devarajan and T. S. Muthu, Enhanced Storage Optimization System (SoS) for IaaS Cloud Storage - *Proc. of IEEE Int. Conf. on Inve. Systems and Control* (2020) pp. 756. https://doi.org/10.1109/ICISC47916.2020.9171182
13. S. S. Manikandasaran, L. Arockiam, and P. D. Malarchelvi, Int. J. Info. Comp. Security **11**, (2019). https://doi.org/10.1504/IJICS.2019.096846
14. L. P. George, D. I. G. Amalarethinam, and A. S. Chandran, GLEnc Algorithm to Secure Data in Public Cloud Environment - *Proc. of IEEE Int. Conf. on Adv. in Comp. Commu. and Infor. (ICACCI)* (2018) pp. 2018-2022. https://doi.org/10.1109/ICACCI.2018.8554451
15. G. Uma and L. Jayasimman, Int. J. Res. Adv. Tech. (IJRAT) (2019).
16. P. L. Prasanna and L. K. Kumar, IOSR J. Comp. Eng. (IOSR-JCE), **19**, 4 (2017).
17. A. Agarwala, P. Singh, and P. K. Atrey, DICE: A Dual Integrity Convergent Encryption Protocol for Client-Side Secure Data Deduplication - *Proc. of IEEE Int. Conf. on Systems, Man, and Cybernetics* (2017) pp. 2176-2181. https://doi.org/10.1109/SMC.2017.8122942
18. J. K. Periasamy and B. Latha, Neural Comlput. Applic. **32**, 485 (2019). https://doi.org/10.1007/s00521-019-04060-9
19. P. Prajapati and P. Shah, J. King Saud Univ- Comp. Info. Sci. (2020), in press. https://doi.org/10.1016/j.jksuci.2020.10.021
20. J. Li, X. Chen, M. Li, J. Li, P. P.C. Lee, and W. Lou, IEEE Trans. on Parallel and Distrib. Sys. **25**, 1615 (2014). https://doi.org/10.1109/TPDS.2013.284
21. M. Bellare, S. Keelveedhi, and T. Ristenpart, DupLESS: Server-aided Encryption for Deduplicated Storage - *Proc. of the 22nd USENIX Conf. on Sec (SEC'13),* USENIX Association, USA (2013) pp. 179–194.