ENABLING BALANCED DATA DEDUPLICATION IN MOBILE EDGE COMPUTING

[1] YADLAPALLI LAKSHMI DURGA

[2] Mr. Naga Srinivasa Rao

[1]PG STUDENT ,DEPT OF MCA

[2] **Asst. Prof**, Dept of MCA

SREE KONASEEMA BHANOJI RAMARS P.G. COLLEGE (AMALAPURAM)

ABSTRACT

In the mobile edge computing (MEC) environment, edge servers with storage and computing resources are deployed at base stations within users' geographic proximity to extend the capabilities of cloud computing to the network edge. Edge storagesystem(ESS), is comprised by connected edge servers in a specific area, which ensures low-latency services for users. However, high data storage overheads incurred by edge servers' limited storage capacities is a key challenge in ensuring the performance of applications deployed on an ESS. Data deduplication, as a classic data reduction technology, has been widely applied in cloud storage systems. It also offers a promising solution to reducing data redundancy in ESSs. However, the unique characteristics of MEC, such as edge servers' geographic distribution and coverage, render cloud data deduplication mechanisms obsolete. In addition, data distribution must be balanced over edge storage systems to accommodate future data demands, which cannot be undermined by data deduplication. Thus, balanced edge data deduplication (BEDD) must consider deduplication ratio, data storage benefits, and resource balance systematically under the latency constraint. In this article, we model the novel BEDD problem formally and prove its *NP*-hardness. Then, we propose an optimal approach for solving the BEDD problem exactly in small-scale scenarios and a sub-optimal approach to solve large-scale BEDD problems with a theoretical performance guarantee. Extensive and comprehensive experiments conducted on a real-world dataset demonstrate the significant performance improvements of our approaches against four representative approaches.

Keywords: Deduplication, Encryption Technique, Decryption Technique.

I. INTRODUCTION

Cloud providers offer potentially infinite storage space, where users can use as much space as they can and vendors constantly look for techniques which aimed to minimize redundant data (multiple copies) and maximize space savings. To minimize redundant data that is the elimination of multiple copies, we make use of deduplication technique. The most widely adopted technique is Cross-user deduplication. The simple idea behind deduplication is to store duplicate Data only once.

Therefore, if a user wants to upload a data which is already stored in the cloud, the cloud provider will show deduplication not allowed. Deduplication can reduce storage needs by up to90-95% for backup applications [11] and up to 68% in standard file systems [23].Users require the protection of their data and confidentiality along with low ownership costs and flexibility which guarantees through encryption. Unfortunately, deduplication and encryption are two conflicting technologies. While the aim of deduplication is to detect duplicate data and store them only once, the result of encryption is to make two identical data indistinguishable after being encrypted. This means that if data are encrypted by users in a standard way, the cloud storage provider cannot apply deduplication since two identical data will be different after encryption. On the other hand, confidentiality cannot be guaranteed and data are not protected against attackers in cloud storage providers if data are not encrypted by users. convergent encryption.

Convergent encryption is a technique which has been proposed to meet these two conflicting requirements [18], [25], [26] whereby the encryption key is usually the result of the hash of the data. If we want to achieve confidentiality and deduplication at the same time, convergent encryption seems to be a good candidate but unfortunately, it suffers from various well-known weaknesses [15], [24]. Here we mainly focus on deduplication and cloud storage. Cloud computing offers a new way of Information Technology services by re-arranging various resources (e.g., storage, computing) and providing them to users based on their demands.

By linking network resources together, cloud computing provides a big pool of resource. It has desirable properties, such as scalability, elasticity, fault-tolerance, and pay-per-use. Thus, it has become a promising service platform. Data storage service is the most important and popular cloud. Cloud users have some personal or confidential data which will be uploaded to the data center of a Cloud Service Provider and allow it to maintain these data. Since we cannot avoid intrusions and attacks towards sensitive data at CSP, it is prudent to assume that CSP cannot be fully trusted by cloud users.

Moreover, the loss of control over their own personal data leads to high data security risks, es pecially data privacy leakages. The privacy issue has become very serious, Due to the rapid development of data mining and other analysis technologies. Hence, a good practice is to only outsource encrypted data to the cloud in order to ensure data security and user privacy.

But the same or different users may upload duplicated data in encrypted form to CSP, especially for scenarios where data are shared among many users. Although cloud storage space is huge, data duplication greatly wastes network resources, consumes a lot of energy, and complicates data management. The development of numerous services will further make it urgent to deploy efficient resource management mechanisms. Here the practical issue is how to manage encrypted data storage with deduplication in an efficient way. However, current industrial deduplication solutions cannot handle encrypted data. Existing solutions for deduplication suffer from some security weakness i.e. brute-force attacks.

They cannot flexibly support data access control and revocation at the same time. Most existing solutions cannot ensure reliability, security, and privacy. In practice, it is hard to allow data holders to manage deduplication due to a number of reasons. First, they cloud cause storage delay because data holders may not be always online or available for such a management. Second, deduplication could become too complicated in terms of communications and computations to involve data holders into deduplication process. Third, in process of discovering duplicated data, it may intrude the privacy. Forth, a data holder may have no idea about how to issue data access rights or deduplication keys to a user in some situations when it does not know other data holders due to data super distribution. Therefore, CSP cannot cooperate with data holders on data storage deduplication in many situations. The results show the superior efficiency and effectiveness of the scheme for potential practical deployment, especially for data deduplication in cloud storage.

II. LITERATURE SURVEY

Encrypted Data Deduplication

Cloud storage service providers such as Dropbox [2], Google Drive [3], Mozy [4], and others perform deduplication to save spa ce by only storing one copy of each file uploaded. However, if clients conventionally encrypt their da ta, storage savings by deduplication are totally lost. This is because the encrypted data are saved as different contents by applying different encryption keys. Existing industrial solutions fail in encrypted data deduplication. For example, DeDu [17] is an efficient deduplication system, but it cannot handle encrypted data. Reconciling deduplication and client -side encryption is an active research topic [1]. Message-Locked Encryption (MLE) intends to solve this problem [5].

The most prominent manifestation of MLE is Convergent Encryption (CE), introduced by Douceur et al. [6] and others [7], [11], [12]. CE was used within a wide variety of commercial and research storage service systems. Letting M be a file's data, a client first computes a key K HðMÞ by applying a cryptographic hash function H to M, and then computes ciphertext C EðK; MÞ via a deterministic symmetric encryption scheme. A second client B encrypting the same file M will produce the same C, enabling deduplication. However, CE is subject to an inherent security limitation, namely, susceptibility to offline brute force dictionary attacks [13], [14]. Knowing that the target data M underlying the target ciphertext C is drawn from a dictionary S ¼ fM1; . . .;Mng of size n, an attacker can recover M in the time for n ¼ jSj off-line encryptions: for each i ¼ 1; . . . ; n, it simply CEencrypts Mi to get a ciphertext denoted as Ci and returns Mi such that C ¼ Ci.

This works because CE is deterministic and keyless. The security of CE is only possible wh en the target data is drawn from a space too large to exhaust. Another problem of CE is that it is not flexible to support data access control by data holders , especially for data revocation process, since it is impossible for data holders to generate the same new key for data re-encryption. An image deduplication scheme adopts two servers to achieve verifiability of deduplication. The CE-based scheme described in combines file content and user privilege to obtain a file token with token unforgeability. However, both schemes directly encrypt data with a CE key, thus suffer from the problem as described above. To resist the attack of manipulation of a data identifier, Meye et al. proposed to adopt two servers for intra-user deduplication and inter deduplication. The ciphertext C of CE is further encrypted with a user key and transferred to the servers. However, it does not deal with data sharing after deduplication among different users. ClouDedup also aims to cope with the inherent security exposures of CE, but it cannot solve the issue caused by data deletion. A data holder that removes the data from the cloud can still access the same data since it still knows the data encryption key if the data is not completely removed from the cloud. Bellare et al. [1] proposed DupLESS that provides secure deduplicated storage to resist brute force attacks.

In Dup- LESS, a group of affiliated clients (e.g., company employees) encrypt their data with the aid of a Key Server (KS) that is separate from a Storage Service (SS). Clients authenticate themselves to the KS, but do not leak any information about their data to it. As long as the KS remains inaccessible to attackers, high security can be ensured. Obviously, Dup - LESS cannot control data access of other data users in a flexible way. Alternatively, a policy-based deduplication proxy scheme [15] was proposed but it did not consider duplicated data management (e.g., deletion and owner management) and did not evaluate scheme performance. As stated in, reliability, security, and privacy should be taken into considerations when designing a deduplication system.

The strict latency requirements of primary storage lead to the focus on offline deduplication systems. Recent studies proposed techniques to improve restore performance. Fu et al. proposed History - Aware Rewriting (HAR) algorithm to accurately

identify and rewrite fragmented chunks, which improved the restore performance. Kaczmarczyk et al. focused on inter-version duplication and proposed Context-Based Rewriting (CBR) to improve the restore performance for latest backups by shifting fragmentation to older backups.

Another work even proposed to forfeit deduplication to reduce the chunk fragmentation by container capping. In our previous work, we proposed using PRE for cloud data deduplication. This scheme applies the hash code of data to check ownersh ip with signature verification, which is unfortunately insecure if HðMÞ is disclosed to a malicious user. A new ownership verificat ion approach to improve our previous work and aim to support data deduplication in an efficient way.

DATA OWNERSHIP VERIFICATION AND OTHERS

Halevi et al. first introduced the practical implementation of Proofs of Ownership (PoW) based on Merkle tree for deduplicati on, which realized client-side deduplication. They proposed to apply an erasure coding or hash function over the original file first and then use Merkle tree on the preprocessed data to generate the verification information. When challenging a prover, a verifier randomly chooses several leaves of the tree and obtains the corresponding sibling - paths of all these leaves. Only when all paths are valid, will the verifier accept the proof.

This construction can identify deduplication at a client to save network bandwidth and guarantee that the client holds a file rather than some part. Pietro et al. chose the projection of a file onto some randomly selected bit-positions as proof to realize the PoW. But both schemes above do not pay attention to data privacy and the server for data storage could be aware of the file c ontent. Ng etal. Adapted the PoW to manage the deduplication of encrypted data. This scheme also generates verification information for deduplication check based on Merkle trees. Each leaf value is generated based on several data blocks, while each interactive proof protocol can only challenge one leaf of the Merkle tree. In order to achieve higher security by checking more data, the proto col should be executed multiple times which introduces much overhead. By relying on dynamic spot checking, a data holder only nee ds to access small but dynamic portions of the original file to generate the proof of possession of the original file, thus greatly reducing the burden of computation on the data holder and minimizing the communication cost between the data holder and CSP.

At the same time, by utilizing dynamic coefficients and randomly chosen indices of the original files, the scheme mixes the randomly sampled portions of the original file with the dynamic coefficients to generate the unique proof in every challe nge. The work focuses on ownership proof of the uploaded data during data deduplication. Use of Eillptic Curve Cryptography (ECC) to verify data ownership with the support of an authorized party. Yuan and Yu attempted to solve the issue of supporting efficient and secure data integrity auditing with storage deduplication for cloud storage.

They proposed a novel scheme based on techniques including polynomial-based authentication tags and homomorphic linear authenticators. Their design allows deduplication of both files and their corresponding authentication tags. Data integrity auditing and storage deduplication are achieved simultaneously. Public auditing and batch auditing are both suppo rted. But the feasibility of supporting deduplication data was not discussed in this work. In order to reduce workloads due to duplicate files, Wu et al. proposed Index Name Servers (INS) to manage not only file storage, data deduplication, optimized node selection, and s erver load balancing, but also file compression, chunk matching, real-time feedback control, IP information, and bus y level index monitoring. To manage and optimize storage nodes based on a client-side transmission status by the proposed INS, all nodes must elicit optimal performance and offer suitable resources to clients.

In this way, not only can the performance of a storage system be improved, but the files can also be reasonably distributed, decreasing the workload of the storage nodes. However, this work cannot deduplicate encrypted data. Fan et al. proposed a hyb rid data deduplication mechanism that provides a practical solution with partial semantic security. This solution supports deduplication on plaintext and cipher text. But this mechanism cannot support encrypted data deduplication very well. It works based on the assumption that CSP knows the encryption key of data. Thus it cannot be used in the situation that the CSP cannot be fully trusted by the data holders or owners. Apply ECC, PRE, and symmetric encryption to deduplicate encrypted data. Our scheme can resist the attacks mentioned above in CE and achieve good performance without keeping data holders online all the time. Meanwhile, it also ensures the confidentiality of stored data and supports digital rights management. We aim to achieve deduplication on en crypted data in the cloud.

III. EXISTING SYSTEM

Cloud storage service providers such as Dropbox, Google Drive, Mozy, and others perform deduplication to save space by only storing one copy of each file uploaded. However, if clients conventionally encrypt their data, storage savings by deduplication are totally lost. This is because the encrypted data are saved as different contents by applying different encryption keys. Existing industrial solutions fail in encrypted data deduplication. For example, DeDu [17] is an efficient deduplication system, but it cannot handle encrypted data. The existing system introduces a model for provable data possession(PDP) that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it.

The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage systems. We present two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees.

In particular, the over head on the server is low (or even constant), as opposed to linear in the size of the data.

Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation. We define a model for provable data possession (PDP) that provides probabilistic proof

that a third party stores a file. The model is unique in that it allows the server to access small portions of the file in generating the proof; all other techniques must access the entire file. Within this model, we give the first provably -secure scheme for remote data checking. The client stores a small O(1)amount of metadata to verify the server's proof. Also, the scheme uses O(1) bandwidth1. The challenge and the response are each slightly more than 1 Kilobit. We also present a more efficient version of this scheme that proves data possession using a single modular exponentiation at the server, even though it provides a weaker guarantee. Both schemes use homomorphic verifiable tags. Because of the homomorphic property, tags computed for multiple file blocks can be combined into a single value.

The client pre-computes tags for each block of a file and then stores the file and its tags with a server. At a later time, the client can verify that the server possesses the file by generating a random challenge against a randomly selected set of file blocks. Using the queried blocks and their corresponding tags, the server generates a proof of possession. The client is thus convinced of data possession, without actually having to retrieve file blocks. The efficient PDP scheme is the fundamental construct underlying an archival introspection systemthat we are developing for the long-term preservation of Astronomy data. We are taking possession of multi-terabyte Astronomy databases at a University library in order to preserve the information long after the research projects and instruments used to collect the data are gone.

The database will be replicated at multiple sites. Sites include resource-sharing partners that exchange storage capacity to achieve reliability and scale. As such, the system is subject to freeloading in which partners attempt to use storage resources and contribute none of their own. The location and physical implementation of these replicas are managed independently by each pa rtner and will evolve over time. Partners may even outsource storage to third -party storage server providers. Efficient PDP schemes will ensure that the computational requirements of remote data checking do not unduly burden the remote storage sites. In a proof -of-retrieve ability system, a data storage center convinces a verifier that he is actually storing all of a client's data. The central challenge is to build systems that are both efficient and provably secure {that is, it should be possible to extract the client's data from any prove that passes a verification check. In this paper, we give the first proof-of-retrieve ability schemes with full proofs of security against arbitrary adversaries in the strongest model, that of Juels and Kaliski. Our first scheme, built from BLS signatures and secure in the random oracle model, has the shortest query and response of any proof-of-retrieve ability with public verifiability.

Our second scheme, which builds elegantly on pseudorandom functions (PRFs) and is secure in the standard model, has the shortest response of any proof-of-retrieve ability scheme with private verifiability (but a longer query). Both schemes rely on homomorphic properties to aggregate a proof into one small authenticator value. For a scalable solution, the amount of comput ation and block accesses at the server should be minimized, because the server may be involved in concurrent interactions with many clients. We stress that in order to minimize bandwidth, an efficient PDP scheme cannot consist of retrieving entire file bloc ks. While relevant, the computation complexity at the client is of less importance, even though our schemes minimize that as well. To meet these performance goals, our PDP schemes sample the server's storage, accessing a random subset of blocks. In doing so, the PDP schemes provide a probabilistic guarantee of possession; a deterministic guarantee cannot be provided without accessing all blocks.

In fact, as a special case of our PDP scheme, the client may ask proof for all the file blocks, making the data possession guarantee deterministic. Sampling proves data possession with high probability based on accessing few blocks in the file, which radically alters the performance of proving data possession. Interestingly, when the server deletes a fraction of the file, t he client can detect server misbehavior with high probability by asking proof for a constant amount of blocks, independently of the total number of file blocks. Our solutions for PDP fit this model: They incur a low (or even constant) overhead at the server and require a small, constant amount of communication per challenge. Key components of our schemes are the homomorphic verifiable tags. They allow verifying data possession without having access to the actual data file.

Disadvantages

- A deterministic guarantee cannot be provided without accessing all blocks.
- It does not provide a data possession guarantee.
- It cannot offer both sampling across blocks and constant storage on the client.
- The security of the scheme is not proven and remains in question.
- Source authentication techniques do not apply to provable data possession.

IV. PROPOSED WORK

We propose a scheme to deduplicate encrypted data at CSP by applying PREto issue keys to different authorized data holders b ased on data ownership challenge. It is applicable in scenarios where data holders are not available for deduplication control. In this paper, aiming at achieving data integrity and deduplication in the cloud, we propose two secure systems namely SecCloud and SecCloud+. SecCloud introduces an auditing entity with a maintenance of a MapReduce cloud, which helps clients generate data tags before uploading as well as audit the integrity of data having been stored in the cloud. Convergent encryption ensures d ata isolation in deduplication. Author formalized this primitive as message locked encryption and explored its application in space efficient secure out sourced storage. The systemalso addressed the problemand showed a secure convergent encryption for efficient encryption without considering issues of the key management and blo ck level deduplication. To encrypt a file using convergent encryption, a client computer a cryptographically strong hash of the file content.

The file is then encrypted using this hash value as a key. The hash value is then encrypted using the public keys of all authorized readers of the file and these encrypted values are attached to the file as metadata. Convergent encryption enables identical encrypted files to be recognized as identical, but there remains the problem of performing this identification across a

large number of machines in a robust and decentralized manner. Cloud User: A cloud user is which who wants to outsource data on public storage which acts as a public cloud in cloud computing. A system provides authenticate used to enter in system upload data with a particular set of privileges for further accessing the uploaded data to download. Public Storage: Public Storage is a storage disk which allows storing the user's data on it's with include of authorized and not allow to upload the duplicate data.

Thus save storage space and bandwidth of transmission. This uploaded data is in encrypted form, only a user with the respective key can decrypt it. Private Cloud: A private cloud acts as a proxy to allow both data owner and user to sec urely perform duplicate check along with differential privileges. Auditor: The set of privileges and the symmetric key for each privilege is assigned and stored in private cloud. The user registers into the system, privileges are assigned to the user according to identity given by the user at registration time; means on basis of post which accesses by the user. The data owner with a privilege issued set wants to upload and share a file to users, further the data owner performs identification and sends the file tag to the private server. Private cloud server verifies the data owner and computes the file token and will send back the token to the data owner. The data own er sends this file token and a request to upload a file to the storage provider. If the duplicate file is found the n user needs to run the PoW protocol with the storage provider to prove he/she has an ownership of the respective file.

In the PoW result; if proof of ownership of the file is passed then the user will be provided a pointer for that file. And on the second case; for no duplicate is found for the file, the storage provider will return a signature for the result of that proof for the particular file. To upload file user sends the privilege set as well as the proof of the private cloud services as a request. The private cloud server verifies the signature first on receiving the request for the user to upload the file. If the result of signature verification is passed, private cloud will compute the file token with each privilege from the privilege set is given by the user, which w ill be returned to the user. Convergent Encryption gives information secrecy in deduplication. Customers get a convergent key from e ach and every unique data copy and encrypt the unique data copy with the co nvergent key. And also, the customer determines a tag for the unique data copy, which will utilize the tag to recognize duplicate copies. The consideration of the tagging accuracy holds that means if both the data copies are the same, then the tags of the data copies are same. To discover the duplicate copies, the customer first sends the tag to the server to verify if the duplicate copy has been already available. The convergent key and tags are individually evaluated, and tags cannot understand the convergent key to distract the data security. The encrypted data copy and the respective tag will store on the server. The convergent encryption system can be defined by four basic functions. The idea of Authorized Data deduplication was proposed to secure the information security by counting differential benefits of clients in the copy check.

The system additionally exhibited a few new deduplication developments supporting approved copy check in hybrid cloud construction modeling, in which the copy check tokens of documents are created by the private cloud server having private keys. Security examination shows that our plans are secure as far as insider and outsider attacks determined in the proposed security model. As an issue verification of idea, they actualized a model of the proposed approved copy check plan and behavior test bed investigates their model. They indicated that their authorized copy check plan brings about insignificant overhead comparing convergent encryption and system exchange. The thought of authorized information deduplication was proposed to ensure the information security by counting differential benefits of clients in the duplicate copy check.

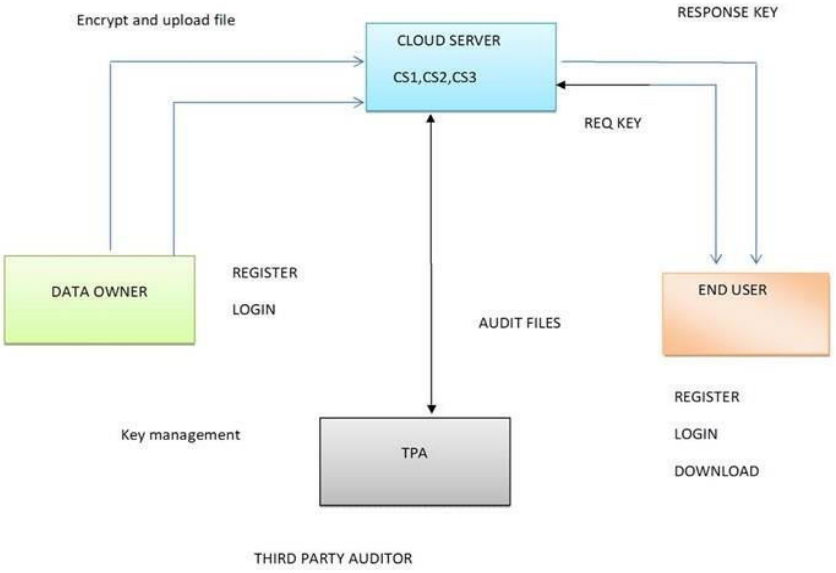


Fig. 4. System model

DATA OWNER

- Register
- Login
- Select the file and send to the cloud server

- If he send the same file to the same cloud server then it should avoid the file to be uploaded
- Duplication will be checked based on inner content of the file

CLOUD SERVER (CS1, CS2, CS3)

- Stores all file details
- Stores all data owner details
- Stores all end user details

END USER

- Request for the SK key
- Enter key and Downloads the file from the cloud server

ADMIN / TPA

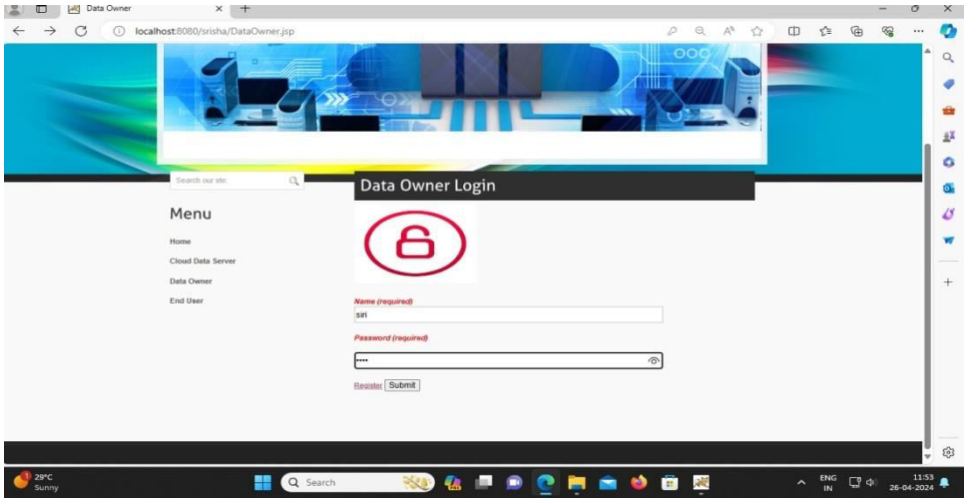
- Audit the file in the cloud server

Advantages

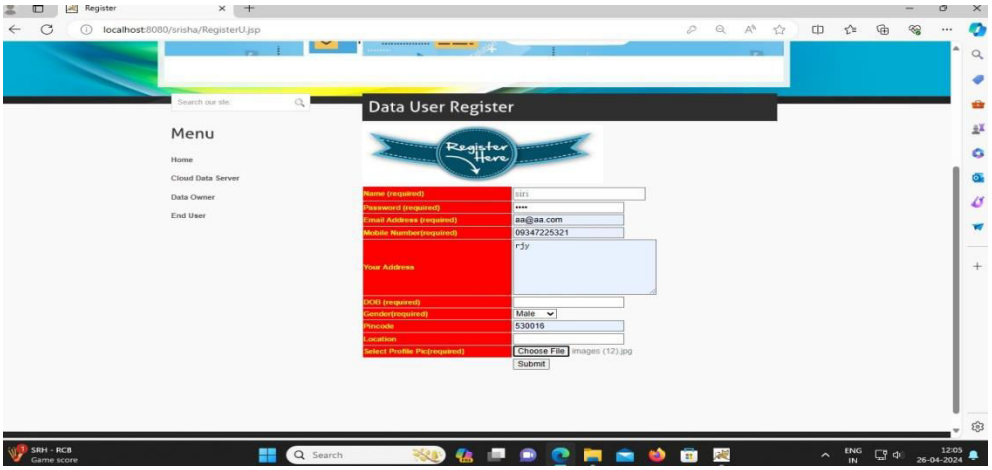
- The client is permitted to perform the duplicate copy check for records selected with the particular subject.
- The complex subject to help stronger security by encoding the record with distinct privilege keys.
- Decrease the storage space of the tags for a reliability check. To strengthen the security of deduplication and ensure the data privacy.
- Any adversary could not directly derive the convergent key from the content of the file and the dictionary attack is prevented.
- Any adversary without the file can-not convince the cloud storage service to get the corresponding access privilege

V. RESULTS

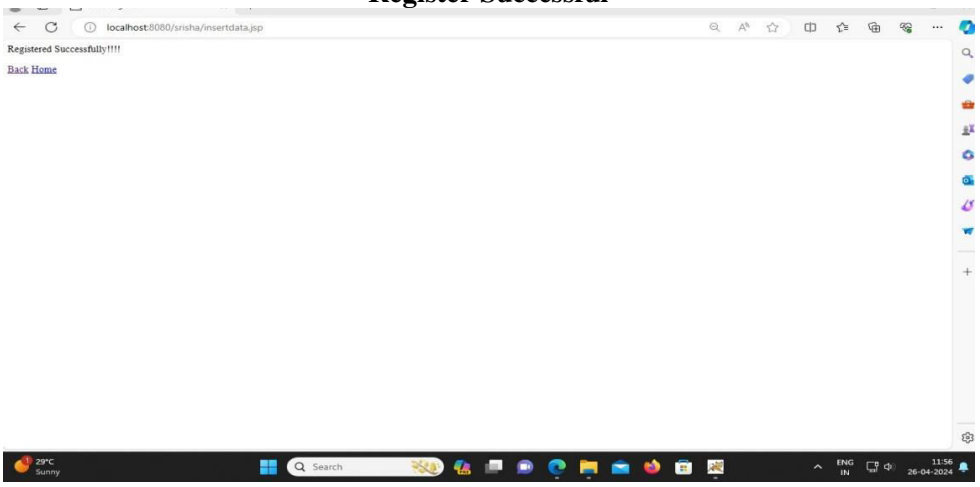
Data Owner Login



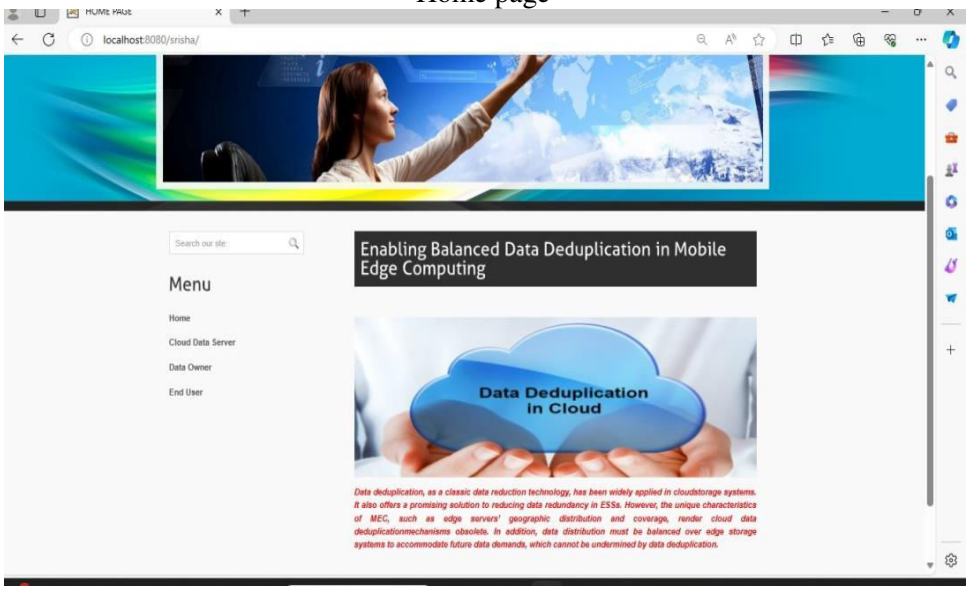
Data Owner Register



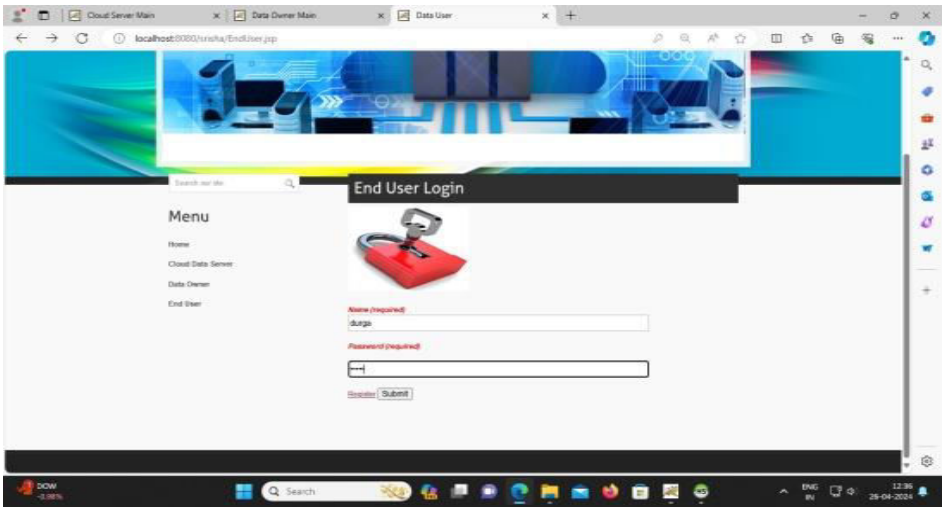
Register Successful

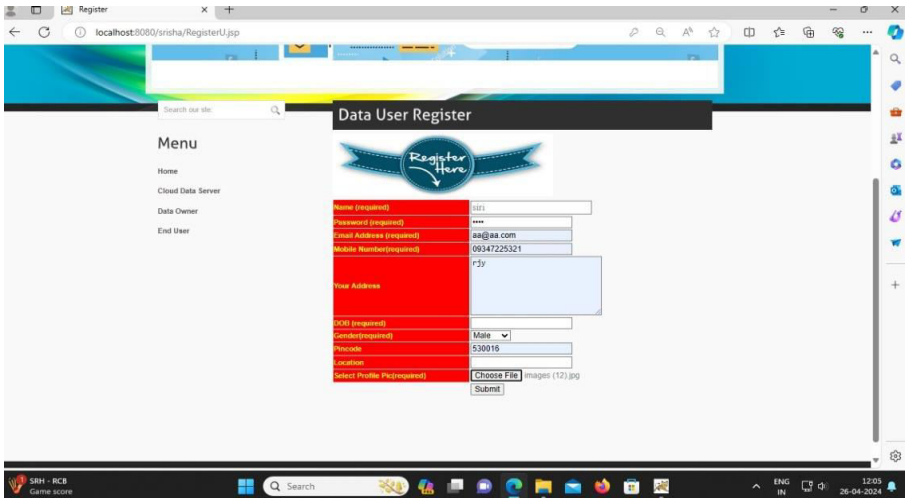


Home page

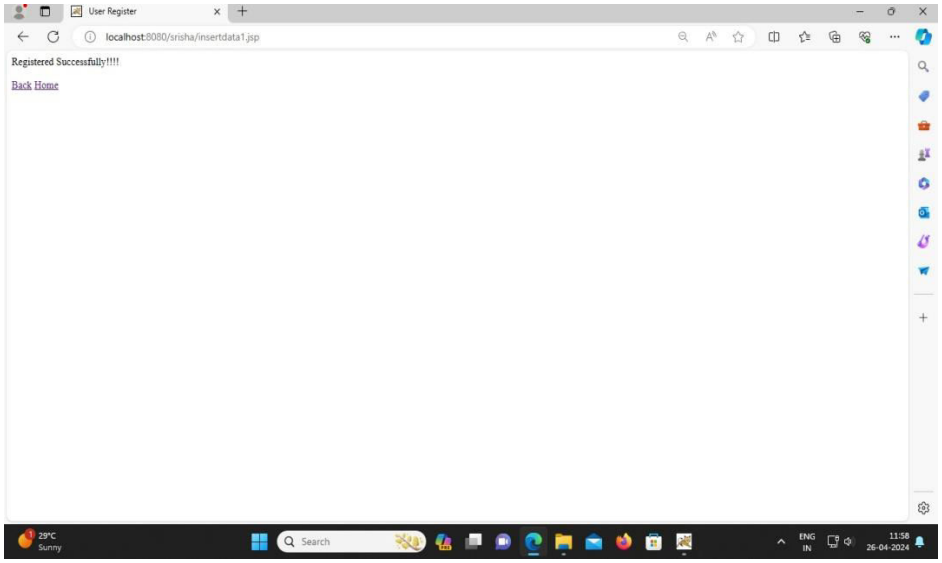


User Login

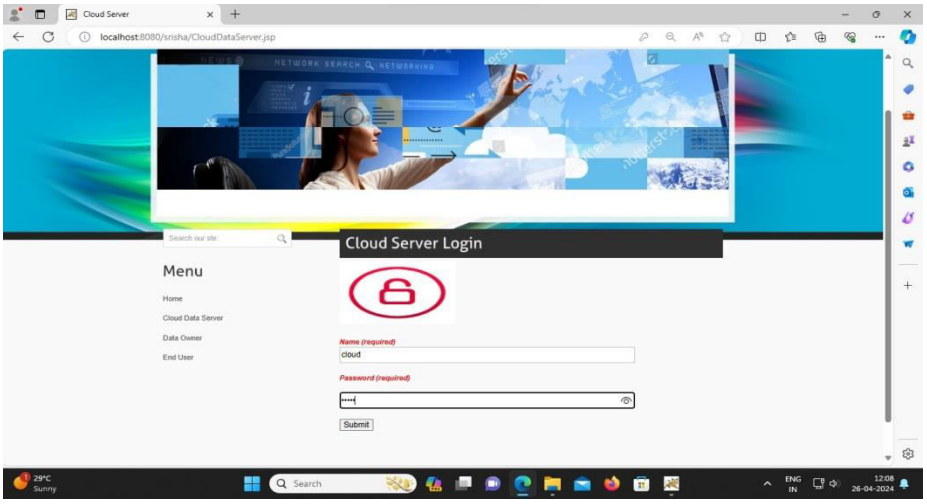




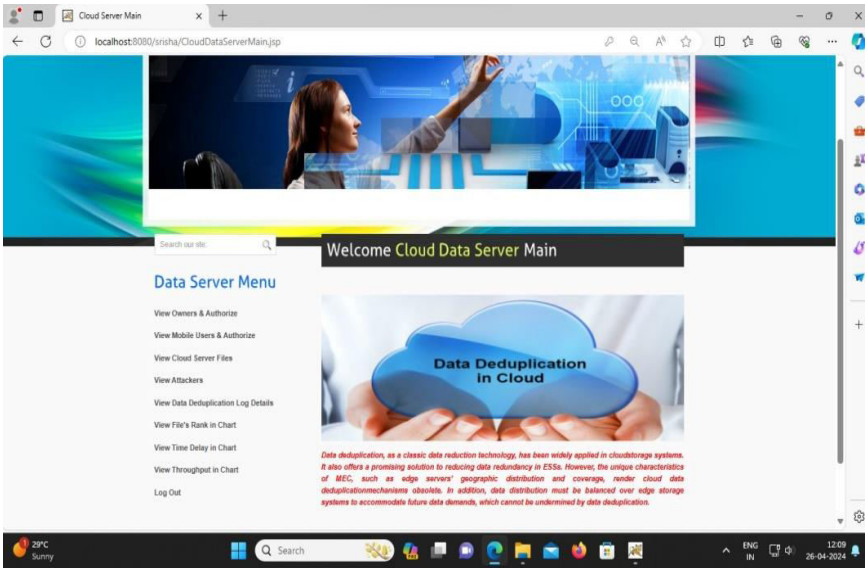
User Register successfully



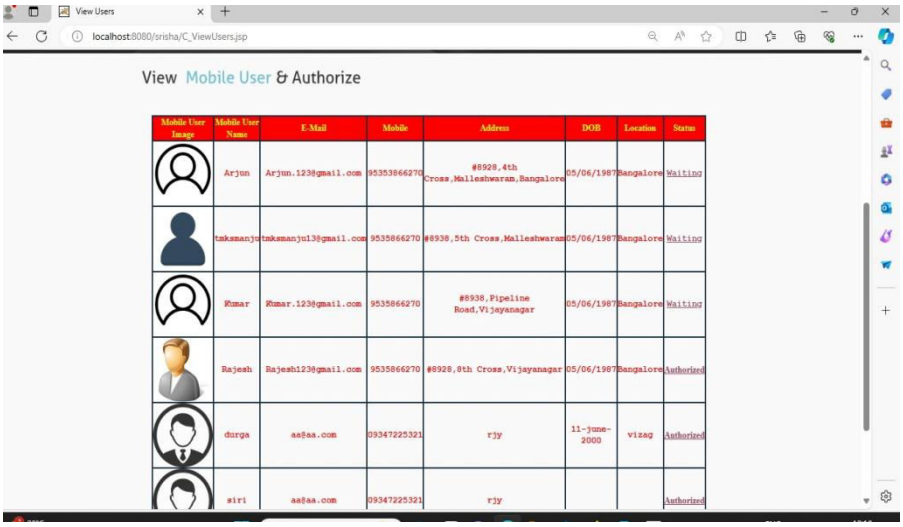
Cloud server login



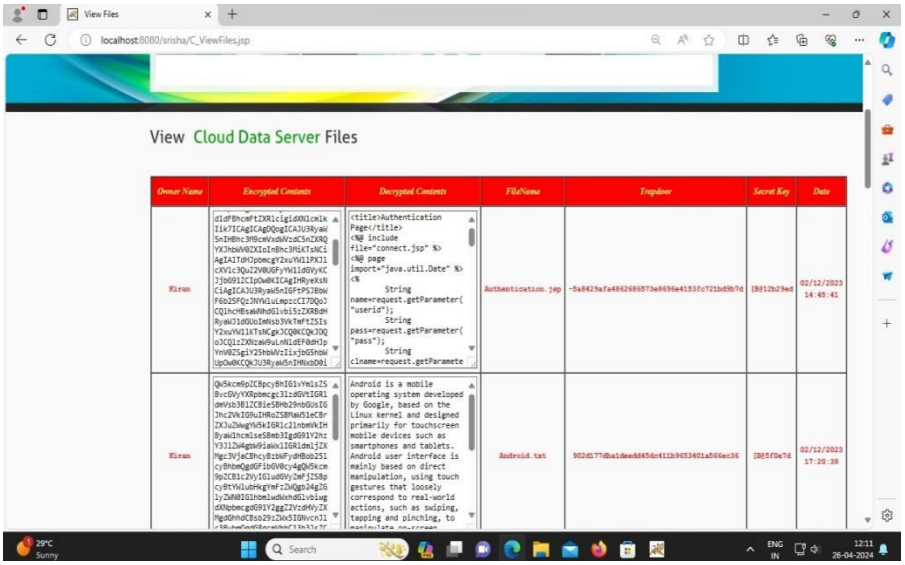
Data server menu



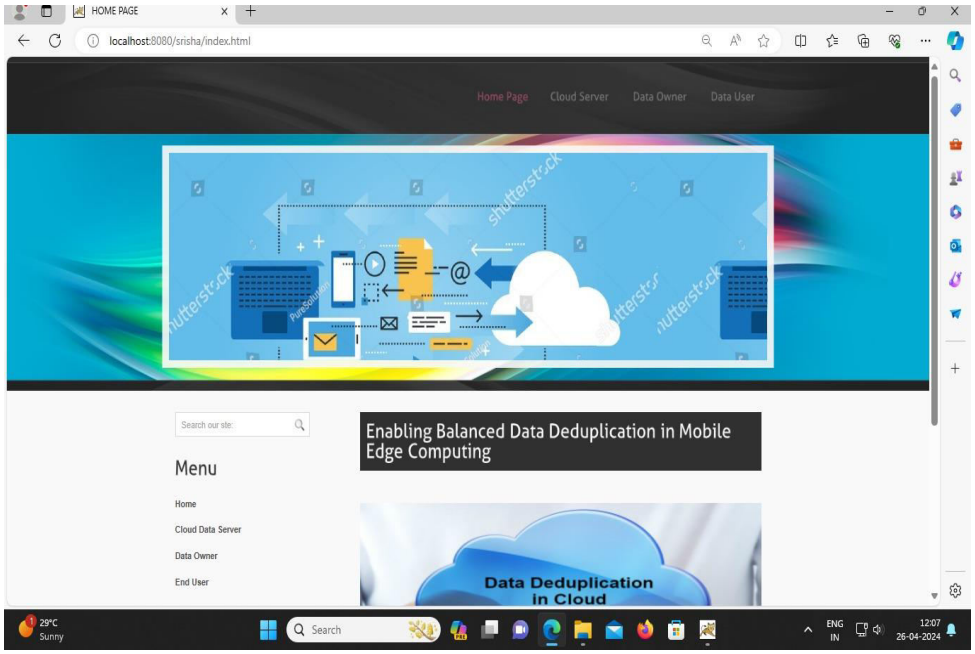
View Mobile User and Authorize



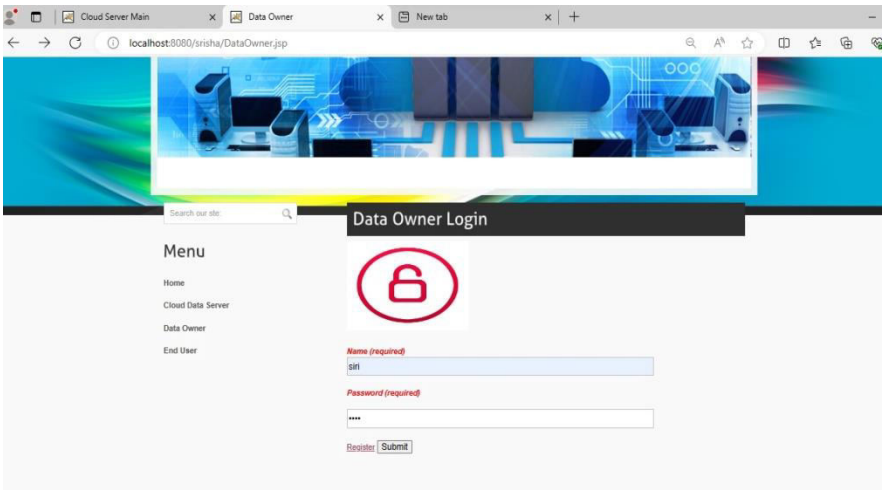
View Cloud Data Server Files



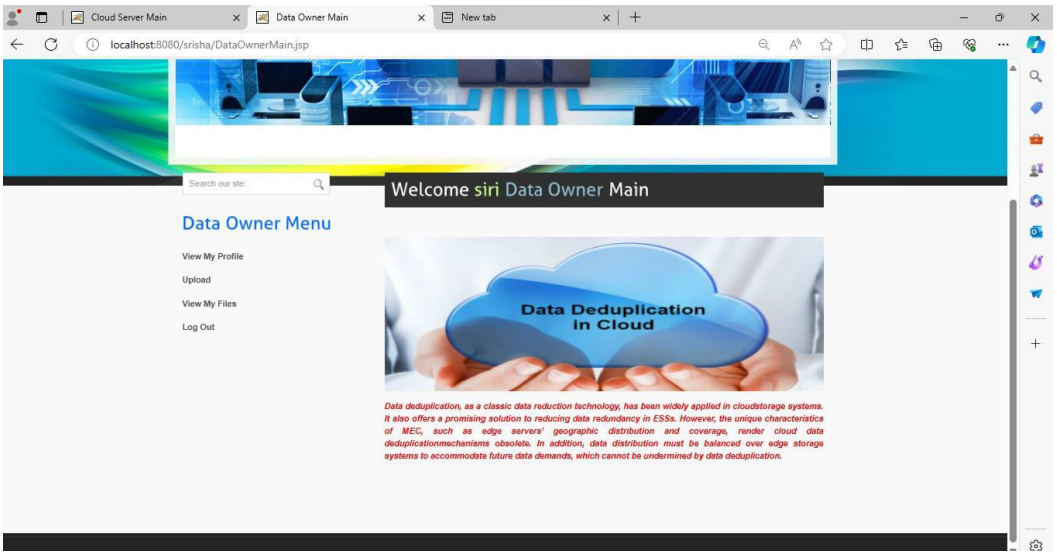
Home page



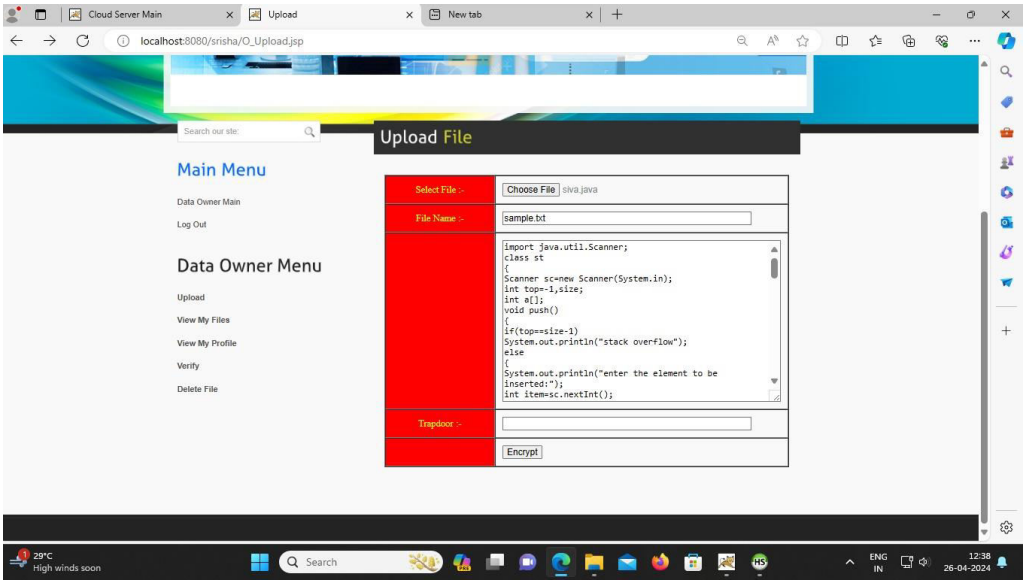
Data Owner Login



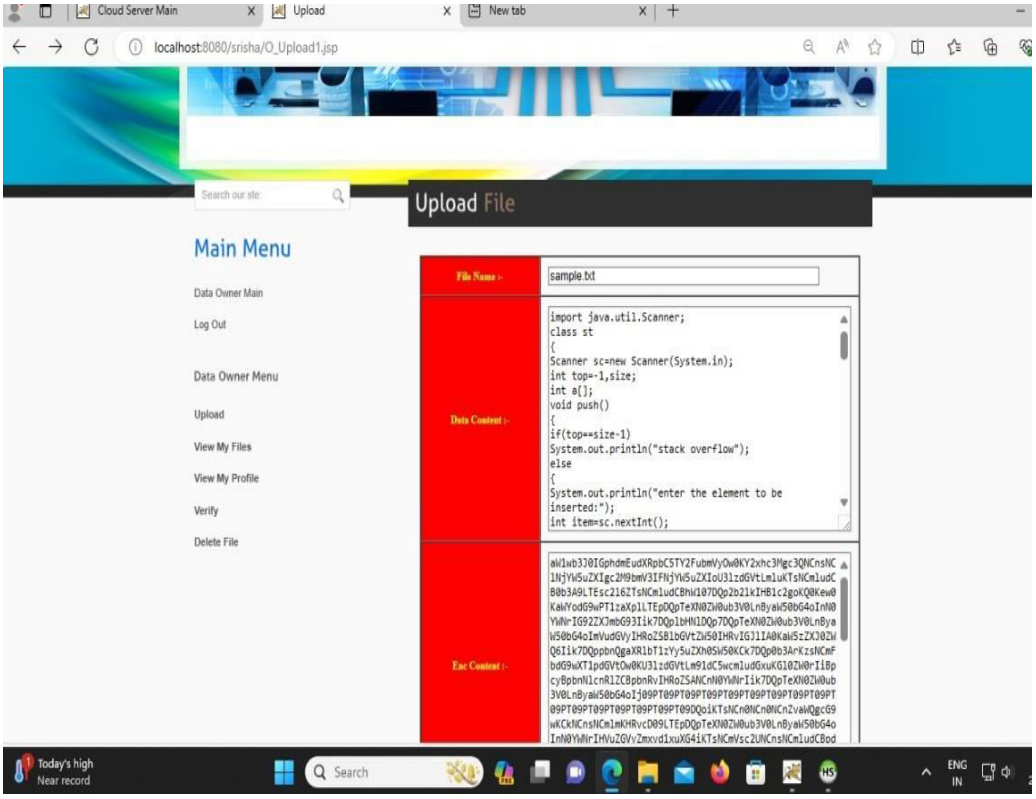
Data Owner Menu


Cloud Server Main
Data Owner Main
New tab
localhost:8080/srisha/DataOwnerMain.jsp
Welcome siri Data Owner Main
Data Owner Menu
View My Profile
Upload
View My Files
Log Out
Data Deduplication in Cloud
Data deduplication, as a classic data reduction technology, has been widely applied in cloudstorage systems. It also offers a promising solution to reducing data redundancy in ESSs. However, the unique characteristics of MEC, such as edge servers' geographic distribution and coverage, render cloud data deduplicationmechanisms obsolete. In addition, data distribution must be balanced over edge storage systems to accommodate future data demands, which cannot be undermined by data deduplication.

Upload File


Cloud Server Main
Upload
New tab
localhost:8080/srisha/O_Upload.jsp
Upload File
Main Menu
Data Owner Main
Log Out
Data Owner Menu
Upload
View My Files
View My Profile
Verify
Delete File
Select File :-
Choose File
siva.java
File Name :-
sample.txt
import java.util.Scanner;
class st
{
Scanner sc=new Scanner(System.in);
int top=-1,size;
int a[];
void push()
{
if(top==size-1)
System.out.println("stack overflow");
else
{
System.out.println("enter the element to be inserted:");
int item=sc.nextInt();
Trapdoor :-
Encrypt
29°C
High winds soon
Search
ENG
IN
12:38
26-04-2024

Encrypt Upload File



VI. CONCLUSION

In this paper, we introduce, motivate, formulate, and solve the balanced edge data de duplication (BEDD) problem, taking into account the data de duplication ratio, data storage benefit, and storage space balance while fulfilling the unique constraints in the MEC environment. We proved its *NP*-hardness and designed two approaches to solve small-scale and large-scale BEDD problems, respectively. Experimental results conducted on a widely used EUA dataset demonstrated the significant performance improvements of our approaches. In the future, we will study the BEDD problem further by taking the network robustness and data reliability into consideration.

REFERENCES

[1] M. Bellare, S. Keelveedhi, and T. Risten part, "Dup LESS: Server aided encryption for deduplicated Storage," in Proc. 22 USENIX Conf. Secur., 2013, pp. 179–194.

[2] T. Y. Wu, J. S. Pan, and C. F. Lin, "Improving Accessing Efficiency of cloud storage using de-duplication and Feedback schemes," IEEE Syst.J., vol. 8, no. 1, pp. 208– 218, Mar. 2014.

[3] Google Drive [Online] Available: http://drive.google.com

[4] Mozy, Mozy: A File-storage and Sharing Service.(2016). [Online]. Available: http://mozy.com/

[5] J. R.Douceur, A. Adya, W. J. Bolosky, D.Simon, And M. Theimer, "Reclaiming space from Duplicate files in a Server less distributed file System, " in Proc. IEEE Int.Conf. Distrib .Comput. Syst., 2002 pp. 617–624,

[6] G. Wallace, et al. ,"Characteristics of backup workloads In production systems," in Proc. USENIX Conf. File Storage Technol., 2012, pp. 1–16.

[7] Z.O.Wilcox, K.Fu "Convergent encryption Reconsidered," Online: Available:http://www.mailarchive.com/cryptograph @ metz dowd.com/msg08949.html,2011.

[8] G.Ateniese, KFu, M.Green, and S.Hohe berger "Improved Proxy re encryption schemes with Applications to secure Distributed storage, "ACM Trans. Inform. Syst. Secur, vol. 9,no.1, pp.1– 30, 2006, doi:10.1145/1127345.1127346.

[9] Open dedup. [Online]. Available: http://opendedup.org/ ,2016

[10] D.T. Meyer and W . J . Bolosky," A study of practical deduplication," ACM Trans. Storage, vol.7, no. 4, pp.1-20, 2012,

[11] Opendedup. http://opendedup.org/
[12] The Freenet Project, Freenet, (2016). [Online] Available: https://freenetproject.org/
[13] J.R. Douceur, A. Adya, W.J. Bolosky, D.Simon and M. Theimer,“ Re-claiming space from duplicate files in a server less distributed filesystem ”, in Proc.IEEE Int.Conf. Distrib Comput. Syst., 2002, pp. 617–624,
[14] M. Bellare, S. Keelveedhi, and T. Ristenpart, “Message- Locked encryption and secure deduplication,” in Proc. CryptoloEUROCRYPT 2013, pp. 296–312.
[15] Mihir Bellare, SriramKeelveedhi, and Thomas Ristenpart. Message–locked encryption and Secure deduplication. In *Advances in Cryptology EUROCRYPT 2013*, pages 296– 312. Springer, 2013