Computers, Materials & Continua
DOI:10.32604/cmc.2022.021797
Article

Tech Science Press

Implementation and Validation of the Optimized Deduplication Strategy in Federated Cloud Environment

Nipun Chhabra*, Manju Bala and Vrajesh Sharma

I. K. Gujral Punjab Technical University, Jalandhar, 144603, India
*Corresponding Author: Nipun Chhabra. Email: nipunchhabra5@gmail.com
Received: 15 July 2021; Accepted: 16 August 2021

Abstract: Cloud computing technology is the culmination of technical advancements in computer networks, hardware and software capabilities that collectively gave rise to computing as a utility. It offers a plethora of utilities to its clients worldwide in a very cost-effective way and this feature is enticing users/companies to migrate their infrastructure to cloud platform. Swayed by its gigantic capacity and easy access clients are uploading replicated data on cloud resulting in an unnecessary crunch of storage in datacenters. Many data compression techniques came to rescue but none could serve the purpose for the capacity as large as a cloud, hence, researches were made to de-duplicate the data and harvest the space from exiting storage capacity which was going in vain due to duplicacy of data. For providing better cloud services through scalable provisioning of resources, interoperability has brought many Cloud Service Providers (CSPs) under one umbrella and termed it as Cloud Federation. Many policies have been devised for private and public cloud deployment models for searching/eradicating replicated copies using hashing techniques. Whereas the exploration for duplicate copies is not restricted to any one type of CSP but to a set of public or private CSPs contributing to the federation. It was found that even in advanced deduplication techniques for federated clouds, due to the different nature of CSPs, a single file is stored at private as well as public group in the same cloud federation which can be handled if an optimized deduplication strategy be rendered for addressing this issue. Therefore, this study has been aimed to further optimize a deduplication strategy for federated cloud environment and suggested a central management agent for the federation. It was perceived that work relevant to this is not existing, hence, in this paper, the concept of federation agent has been implemented and deduplication technique following file level has been used for the accomplishment of this approach.

Keywords: Federation agent; deduplication in federated cloud; central management agent for cloud federation; interoperability in cloud computing; bloom filters; cloud computing; cloud data storage

This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

1 Introduction

Cloud computing is a very progressive technology and, it is stretching its wings in all directions due to its most applaudable feature "pay-as-you-use" [1]. It can be broadly classified into three categories viz. Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [2,3]. Infrastructure as a Service utility provides servers and storage on demand, hence deduplication strategies are employed on this service model. With the growing demand of cloud storage, heterogeneous service providers are coming together under one coalition known as Cloud Federation. Cloud Federation allows providers to share resources through Federation Level Agreements (FLA) covered under federation strategy and if any of the providers in federation is not able to meet the requirements, it is outsourced to other CSPs of federation. Moreover, resources which are being less utilized by any service providers can be leased to other federation members, leading to optimal usage of resources [4]. Despite the optimal utilization of storage and resources due to federation, replication of data consumes most of the space that ought to be eliminated to optimize the cloud storage capacity. To manage this issue, researchers presented many deduplication techniques but most of them were not able to provide an optimal solution to harvest waste storage from CSPs. Deduplication in federated clouds is a novel area to explore and has a lot of scope for researchers to render innovative strategies to garner the storage with reduced time for lookup operations. Third-party agents and brokers provide effective monitoring services that ensure that customers are getting the most out of their cloud environment.

This paper is structured into various sections for better presentation of the study. Section 2 entails a detailed literature survey over the advancement of cloud federation and potential of implementing deduplication in it. Section 3 presents the research gap and existing system while Section 4 confers the proposed system by implementing the concept of federation agent in optimized deduplication approach for federated cloud environment. In Section 5, observations have been recorded and discussions on results have been conducted. Section 6 concludes the paper and sheds light on the future scope of this work.

2 Literature Survey

Recently, many researches are being conducted on interoperability in cloud computing and various deployment models under one coalition to make the optimal usage of the cloud resources. This literature survey has been divided into two sections to understand the progressive metamorphosis of cloud federation and deduplication.

2.1 Progression in Cloud Federation

In the modern context, having limited physical resources is the primary limitation of clouds. If a cloud has exhausted all the computational and storage resources, it cannot provide the service to its clients [5]. All such situations can be addressed by the Inter-Cloud for providing computational service, storage, or any kind of resources to the other clouds. Author in [6] proposed and established Service Level Agreement (SLA) between the service providers and consumers for accessing the resources in cloud federation. Federated cloud plays a major role and establishes a business deal where service provider can sell resources. Authors in [7] rendered Two-level Federations i.e., horizontal level and vertical level, where federation taking place at one level of the cloud is called horizontal level while vertical federation spans across multiple levels. Authors in [8] deliberated the prerequisites for federation architecture as Interface standards and a service broker which acts as a liaison between various standards and interfaces of CSPs. Author in [9] presented a novel concept for establishing a single common broker for all clouds participating in a federation. This acts a third party regulatory body that checks the belongingness of the cloud to the federation. Authors in [10] have defined the federation to be inter-clouds organization which should be dispersed geographically and must have a structured marketing system based on federal SLAs. Authors in [11] have termed components of this architecture as Cloud Coordinator, the Cloud Broker, and the Concentrator. Cloud Coordinator (CC) is a module that interacts with cloud brokers and other cloud coordinators. In the federation, it is the CC that deals with the broker during any event like scheduling of resources and manages the execution, in its domain. Cloud broker mediates between cloud coordinators and consumers and it is based on Service-Oriented Architecture. Its main task it to look for the services available in federated Clouds and employ them for the optimal benefit.

Authors in [12] have projected cloud federation as a mesh of different clouds which are brought together under open standards meant to offer a ubiquitous computing environment. In this decentralized system, policies are well defined under agreeable SLAs that become the foundation for multi-provider infrastructure to act as one entity. Authors in [13] have envisioned for the popularity of cloud computing to be essential for day to day life. In this paper [14] for managing, creating and using identities in the cloud infrastructure a provision has been proposed and named as Federated Cloud Identity management where an application or user is recognized by its credentials. Authors in [15] proposed the management of cloud storage which is heterogeneous in nature and propounds deduplication management and access control. It proves to be a very flexible scheme and can adapt to different scenarios and establishes economic principles.

2.2 Data Deduplication in Cloud

With the increased hand held electronic devices in the market and pervasive internet access, gigantic amount of data is being generated which is stored at various cloud storages. Mostly it is the same data being replicated at and from various locations. Even the regular backups which are maintained at organizations and the identical files created by different departments of an organization lead to data replication which further results in wastage of storage space. The technique through which only a single instance of data is stored while a logical references of this single copy of data is passed as and when required in future, to avoid the storage wastage, is known as deduplication. Categorization of deduplication techniques is done on the basis of time, location and granularity. File level or block level deduplication is observed under granularity based deduplication. Source side and destination side deduplication are categorised under location based and deduplication can be post process or inline process if implemented in reference to time [16]. For data access control in cloud environment [17] authors have proposed to employ client side encryption for the data security across the network. It was mentioned that with the increased amount of data being stored at cloud storages, to perform deduplication on encrypted data was not possible and the solution fall flat. To overcome this issue Message Locked Encryption technique [18] came out as a solution and subsequently it evolved into Convergent Encryption. As CE suffered from Brute force attack in [19] authors have further proposed DupLESS technique in which users encrypt their data using the keys obtained from a Key Server (KS). Some data owners don’t let third party (KS) to manage their data. Splitting of CE and sharing it into multiple CSPs was presented by Li et al. [20]. Cross-user deduplication scheme was proposed by authors in [21] which involved client-side encryption by applying a password authenticated key exchange protocol without any additional independent servers. Secure data deduplication technique was proposed by authors in [22] where AES algorithm and MD5 were used. Hybrid data deduplication mechanism was proposed by Fan et al. [23], where both storage and bandwidth utilization can be improved by source-based approach. For confidentiality of data, a secret sharing technique has been proposed by authors in [24] to encode fragmented data. This secret share shall be granted to the authorized users only that can claim the ownership of the data. Content-identified convergent encryption technique was proposed by authors in [25]. Client controlled deduplication scheme was proposed in [26] that addresses the issues in deduplication from client side. An advanced technique for lookup operations was proposed by Burton Howard that used a probabilistic data structure for checking the membership of an element. This time and space efficient method does not yield any false negatives but may return some false positives [27]. Authors in [28] presented a review on various data deduplication techniques. From the deep study of technical literature, it was observed that with the increased usage of computing devices the plead of storage and computing resources on cloud increased exponentially, which brought many CSPs together to form a federation of clouds. With this increased storage space, users started storing several copies of data that made deduplication a necessity to garner duplicated storage. Moreover, the above schemes cannot flexibly manage data deduplication in various situations and across multiple CSPs.

3 Problem Formulation and Research Gap

3.1 Research Gap

The literature survey revealed that many policies have been devised for deduplication in public and private cloud deployment models with various hashing techniques. Owing to the need for load sharing in cloud computing, interoperability has included many CSPs which are bound by Federation Level Agreements (FLA), either private or public under an alliance known as Federated Cloud. Therefore, during deduplication the search for duplicate copies is made in all the participating CSPs, whether private group or public group, instead of single group. Also, in the optimized deduplication technique in federated clouds, due to the different nature of CSPs, a single file is being stored at private group as well as public group in the same cloud federation which can be further optimized if the central management of the federation is possible. Deep study of technical literature suggests that it has a lot of potential for further research, as not much work has been done, leading to a huge research gap in this field and hence there is enough scope for further research.

3.2 Existing System

In the existing system (as shown in Fig. 1), encryption technique is used for access control followed by deduplication using hashing, for lookup operations, in private and public cloud environment.

3.3 Optimized Data Duplication Strategy

Upon assessing the existing system, it was concluded that a better policy for deduplication can be devised by deploying bloom filters for checking the membership of an element in federated cloud environment instead of public/private cloud groups [29]. This probabilistic data structure (bloom filters) proves very beneficial in the look up operation and brings results in relatively shorter time duration than hashing. In this optimized deduplication strategy three public CSPs and two private CSPs were included in one federation and bloom filters were employed for searching a file in the federation. The proposed strategy establishes two principles as given below:

a) The same file can be stored only once from the same user in case of Private CSPs, whereas in Public CSPs it can be stored once only irrespective of the user, as a reference can be passed to all the other entries.

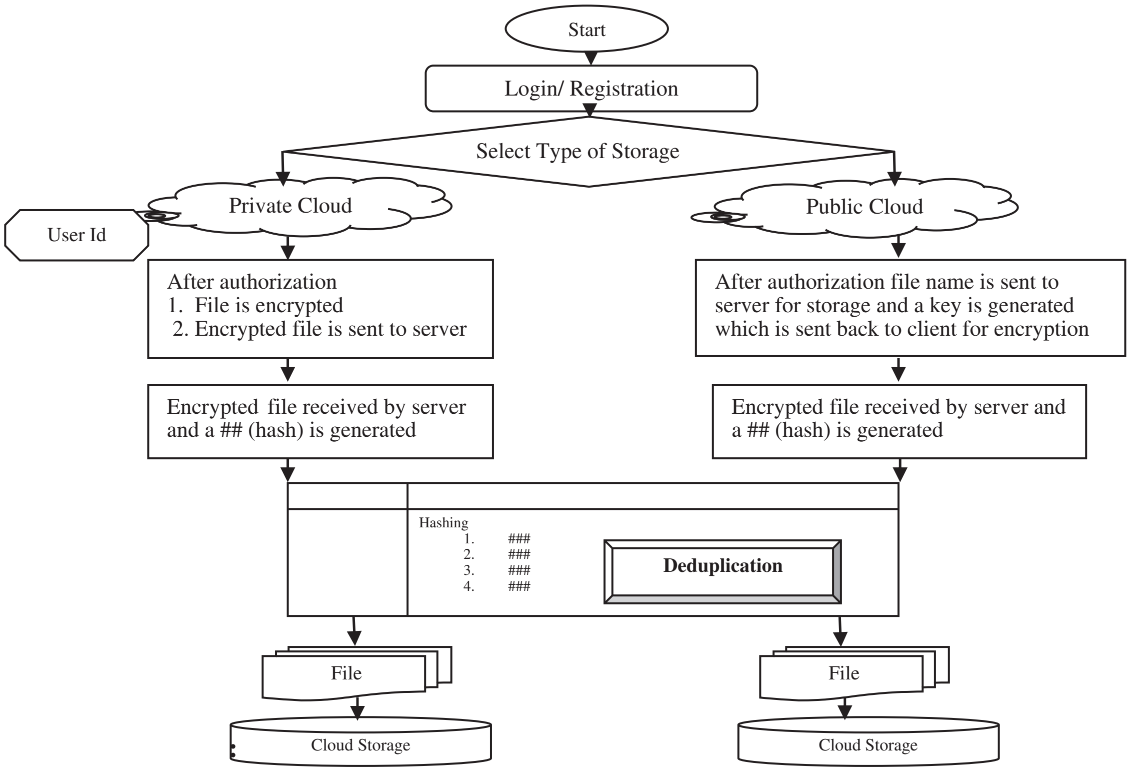

Start
Login/ Registration
Select Type of Storage
Private Cloud
Public Cloud
User Id
After authorization
1. File is encrypted
2. Encrypted file is sent to server
After authorization file name is sent to server for storage and a key is generated which is sent back to client for encryption
Encrypted file received by server and a ## (hash) is generated
Encrypted file received by server and a ## (hash) is generated
Hashing
1. ###
2. ###
3. ###
4. ###
Deduplication
File
File
Cloud Storage
Cloud Storage

Figure 1: Diagrammatic representation of the working of the existing system

b) On deleting a file which has many references in public CSPs, only a logical pointer is deleted for that particular user not the original file.

The proposed optimized deduplication strategy yields excellent results and harvests storage from duplicated data by saving a unique file in Private CSPs and a single copy of a file in Public CSPs. Moreover, by implementing bloom filters, in addition to the space, the total time for lookup operations has also reduced.

4 Proposed System

4.1 Further Optimization of Data Duplication Strategy Through Federation Agent

Though, the previously proposed system (optimized data duplication strategy for federated cloud environment using Bloom Filters) had shown exceptional results and harvested memory from the cloud storage while decreasing the total time during lookup operations, still, this system can be advanced to gain more optimization in deduplication. We have seen that in proposed system one unique file can be saved in Private Clouds group (CSP1 & CSP2) and the same file can also be saved in the Public Clouds group (CSP3, CSP4, CSP5). Therefore, even in the same federation, due to different nature of CSPs, a single file is being stored twice, once in private group and then in public group, in the same cloud federation.

Deep analysis of technical literature suggests that if the central management of the federation is governed by one agent that keeps the check for every entry in any CSP in one federation, this issue can be managed and the agent can be referred as Federation Agent (as shown in Fig. 2). Federation

Agent records every entry in the federation management table and further the lookup operations are performed on this federation management table instead of two separate tables as the case was for the previously proposed system. This can be seen in the Tab. 5 (given ahead) where file 'f1' is stored only one time and for the other times only a reference is passed instead of the saving the same file again and again. The Comparative analysis of the existing system and proposed system has been tabulated in Tab. 1 as given below:

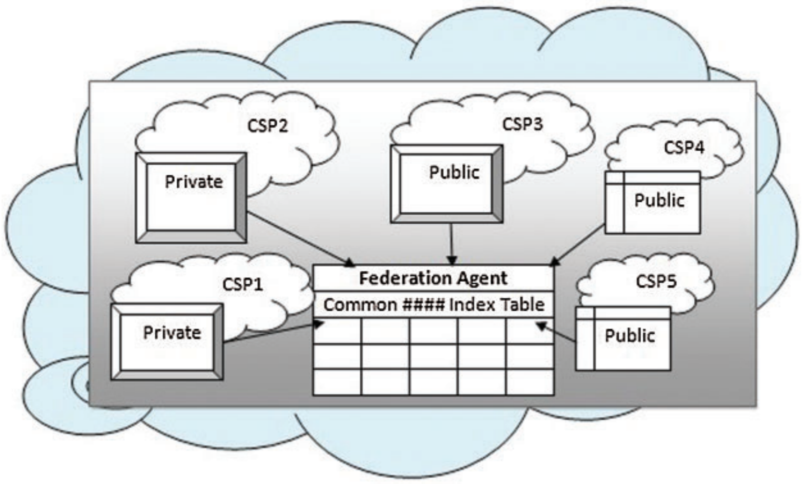


Figure 2: Diagrammatic representation of federated clouds with federation table and federation agent

Table 1: Comparative analysis of existing system and proposed system in tabular form

S. No.	Existing system	Proposed system
1	In the existing system the deduplication has been implemented on either private or public clouds	The deduplication technique has been optimized to accommodate federated cloud environment
2	All the files stored by private users are stored	Only Unique files stored by private users are stored
3	Files by public users are not checked for any entry made by private users	Files by public users are checked for any entry in the Federation table, if it exists, only the reference to the existing file is passed
4	Due to storage of same file in different CSPs the total memory consumption is more and redundancy factor is also high	With the implementation of federation agent (central management agent) total memory consumption and redundancy factor is less than that of the existing system

4.2 Methodology

An optimized data duplication strategy has been proposed and the methodology for the events has been given as below:

1. Simulation is initialized by starting CloudSim 3.0.3 package that creates the datacenter broker, virtual machines, cloudlets and Federation Agent.

Let cloud be a set of elements represented as follows:

cloud= {Dn, Nbw, Dm, Br}

where Dn is number of devices, Nbw is Network bandwidth, Dm is the Deployment Model and Br is the Broker.

Further, Devices be a set of elements represented as,

Devices = {Nodes, Switches, Storage, Controller}

Storage be a set of elements represented as Storage = {fa . . . .fm} where fa to fm are set of files stored in datacenter.

Therefore, federation of clouds can be represented as

FedC = {{Pvt1, Pvt2} {Pub3, Pub4, Pub5}}

where Pvt1 and Pvt2 are Private Clouds and Pub3, Pub4, Pub5 are Public CSPs

For an instance any file (fn) belonging to any User (Un) can be represented as Unfn

private CSP

Hence, Unfn ∈ {{CSP1 U CSP2} || {CSP3 UCSP4 U CSP5}}

where CSP1 and CSP2 are Private CSPs and CSP3, CSP4, CSP5 are Public CSPs

2. Option 1: Private access: When an authorized user signs-in in a chosen Private CSP, its users' credentials are used to encrypt the selected file and this encrypted file is sent to the server.

After Logging in by user U1, a key is generated based on his attributes U1 → keyG

A file f1 is selected for uploading and f1 is encrypted with KeyG → (Enc(f1))

The encrypted file Enc(f1) is transferred to cloud server and then a hash value is computed Hash(Enc (f1))

$$\mathrm{Enc}\,(f1) \rightarrow (\{\mathrm{CSP1},\ \mathrm{CSP2}\} = \mathrm{Hash}\,(\mathrm{Enc}\,(f1)))$$

Using Bloom Filters for checking membership of an element, BF(). This function contains set of elements from 0 to n−1 where n is length of an array.

FA = Hash (Enc (f1)

For deduplication using Bloom Filters, a lookup operation is conducted against generated hash.

$$\mathrm{found} = \mathrm{Hash}\,(\mathrm{Enc}\,(f1)) \in (\mathrm{BF}\,(\{\mathrm{CSP1},\ \mathrm{CSP2}\}))?1:0$$

```
If found == 0
set.add ( element (f1))
f1 → {CSP1 || CSP2}
Storage = {fa . . . .fm + f1}
Else
f1⊆ {CSP1 || CSP2}
FA.update (&reference (f1))          //'&reference' is passed instead of saving the file
```

Option 2: Public Access: When an authorized user logs-in in the selected Public CSP, the name of the file which is to be uploaded is sent to the server where a key is generated and finally this key is sent back to the client.

The key received at client side is used to encrypt the selected file.

After Logging in by Anonymous (Public) user, a file f1 is selected for uploading

$$f1 \rightarrow (\{CSP3,\ CSP4,\ CSP5\} = KeyH))$$

This KeyH is used to encrypt the file fi and transferred to cloud server where a hash value is computed.

$$(Enc(KeyH(f1))) \rightarrow (\{CSP3,\ CSP4,\ CSP5\} = Hash(Enc(KeyH(f1)))$$

Using Bloom Filters for checking membership of an element, BF() For 0 to n−1 where n = length of the array

For deduplication using Bloom Filters, lookup operation is conducted against generated hash.

```
    Dedup()
    {
found = $Hash\,(Enc\,(KeyH\,(f1))) \in (BF\,(\{CSP3,\ CSP4,\ CSP5\}))?1:0$
    If found == 0
    set.add (element)
    f1 → {CSP3 || CSP4 || CSP5}
    Else
    f1 ⊆ {CSP3 || CSP4 || CSP5}
    '&reference' is updated in the status instead of again saving the file
    CSPk.update (f1)
    Where CSPk may be CSP3, CSP4 or CSP5.
    }
```

3. Federation Agent FA(), which feeds on Bloom Filters BF(), searches the file in the federation table and checks for the replicated file.

The file can be Hash(Enc(f1)) in case of Private CSPs or Hash(Enc (KeyH (f1)) in case of Public CSPs represented as Unfn.

```
    Dedup()
    {
found = $(U_n f_n \in \{FA\,[BF\,(FedC)]\})?1:0$
    If found == 0
    {
    FA.update (Unfn)
    set.add (element)
    Unfn.. → {FedC}
    Unfn.. ⊆ {FedC}
    }
    Else
    Unfn ⊆ FA and '&reference' is updated in the status instead of again saving the file
    }
    4. If the file doesn't exist then a new file is stored otherwise a logical reference is passed.
```

4.3 Pseudocode for the Optimized Data Duplication Strategy Through Federation Agent

```
Step1: Initialize the cloudsim 3.0.3 and create datacenter broker, virtual machines and cloudlets.
Step2: If Private storage                         //Public Storage
           Select CSP1 or CSP2
Step 3: User 'U1' logs in with his credentials.
Step 4: A key 'KeyG' is generated.                //Authentication of user is checked from the server side
Step 5: A file 'f1' is selected for uploading on cloud server.
Step 6: The selected file is encrypted and encrypted code E(f1) is generated using a KeyG.
Step 7: This encrypted file E(f1) is transferred to cloud server and then a hash value is computed
        H(E(f1))
        found= f_agent(H(E(f1)));                 // call function
            If found == 0
            Upload the new file on Cloud storage
            Else
            File already exists
        Else                                      //Public Storage
        Select CSP3/CSP4/CSP5
Step 8: Authentication of user is checked         // user login
Step 9: A file 'f1' is selected and file name is sent to server.
Step 10: A key 'KeyH' is generated for received file //User Authentication is checked from the server side
Step 11: The file is encrypted using the 'KeyH' received from server.
Step 12: This encrypted file E(f1) is transferred to cloud server and a hash value is computed H(E(f1))
         found = f_agent(H(E(f1)));               // call function
         If found == 0
         Upload the new file on Cloud storage
         Else
         File already exists
```

```
Step 13: Exit;
Step 14: Int f_agent(H(E(f1)))                //function Federation agent
          {
               For j = 1 to k                 //k = number of hash value in Federation Table
               {
               If Hj(E(f)) == 1
                    File already present in cloud storage
                    Return 1;
               Else
                    File not present
                    Return 0;
               Next++;
               End For
          }}
```

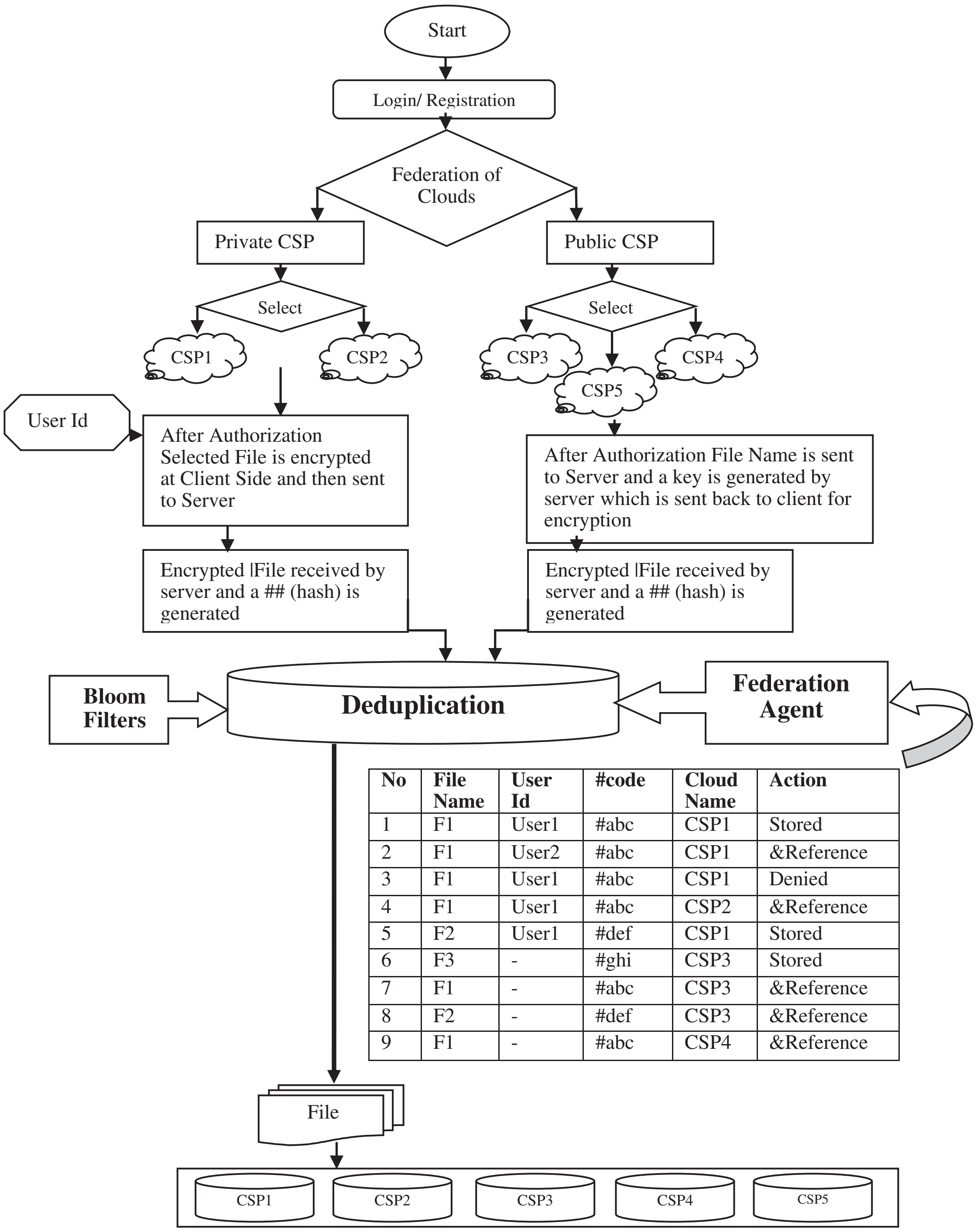
5 Observations

5.1 Simulated Outcomes

Experimental results of the proposed system are stated in Tab. 2 and the diagrammatic representation of the proposed system has been given in the Fig. 3, given below:

Table 2: Actions performed during deduplication in federated clouds in Federated CSPs

Sr. No.	Name of the file	User_Id	#code	Cloud_name	Action_status
1	File1	User1	#abc	CSP1	Stored
2	File1	User2	#abc	CSP1	&Reference
3	File1	User1	#abc	CSP1	Denied
4	File1	User1	#abc	CSP2	&Reference
5	File2	User1	#def	CSP1	Stored
6	File3	–	#ghi	CSP3	Stored
7	File1	–	#abc	CSP3	&Reference
8	File2	–	#def	CSP3	&Reference
9	File1	–	#abc	CSP4	&Reference


Start
Login/ Registration
Federation of Clouds
Private CSP
Public CSP
Select
Select
CSP1
CSP2
CSP3
CSP4
CSP5
User Id
After Authorization Selected File is encrypted at Client Side and then sent to Server
After Authorization File Name is sent to Server and a key is generated by server which is sent back to client for encryption
Encrypted |File received by server and a ## (hash) is generated
Encrypted |File received by server and a ## (hash) is generated
Bloom Filters
Deduplication
Federation Agent
File
CSP1
CSP2
CSP3
CSP4
CSP5

No	File Name	User Id	#code	Cloud Name	Action
1	F1	User1	#abc	CSP1	Stored
2	F1	User2	#abc	CSP1	&Reference
3	F1	User1	#abc	CSP1	Denied
4	F1	User1	#abc	CSP2	&Reference
5	F2	User1	#def	CSP1	Stored
6	F3	-	#ghi	CSP3	Stored
7	F1	-	#abc	CSP3	&Reference
8	F2	-	#def	CSP3	&Reference
9	F1	-	#abc	CSP4	&Reference

Figure 3: Diagrammatic representation of the optimized deduplication strategy, in federated cloud with federation agent

Actions performed during deduplication in federated clouds using Federation Agent as shown in the Tab. 2, are given below:

i. When a user 'User1' tries to upload file1 with hash code #abc for the first time on private cloud service provider 'CSP1', the file1 is uploaded/stored and the status is shown as 'Stored'.
ii. When a user 'User2' tries to upload file1, with hash code #abc, at private cloud service provider CSP1, then instead of saving the same file again the reference to file1 is passed and the status of action is updated as '&Reference'.
iii. When a user 'User1' tries to upload file1, with hash code #abc, at private cloud service provider CSP1, the action status is updated as 'Denied' as the same file already exists at CSP1 with same user (User1).
iv. When a user 'User1' tries to upload file1, with hash code #abc, at private cloud service provider CSP2, then instead of saving the same file again the reference to file1 is passed and the status of action is updated as '&Reference'.
v. When a user 'User1' tries to upload file2, with hash code #def, for the first time at private cloud service provider CSP1, the file2 is uploaded/stored and status of the action is shown as 'Stored'.
vi. When a public user tries to upload file3, with hash code #ghi, for the first time at public cloud service provider CSP3, the file3 is uploaded/stored and status of action is shown as 'Stored'.
vii. When a public user tries to upload file1, with hash code #abc, at public cloud service provider CSP3, then the reference to file1 is passed and the action status as '&Reference' is updated.
viii. When a public user tries to upload file2, with hash code #def, at public cloud service provider CSP3, then the reference to file2 is passed and the action status as '&Reference' is updated.
ix. When a public user tries to upload file1, with hash code #abc, at public cloud service provider CSP4, then the reference to file1 is passed and the action status as '&Reference' is updated.

5.2 *Results and Discussions*

By implementing Federation Agent, which is centrally managing all the transactions in common federation table, it becomes well aware about the existence of any file in any of its participating CSPs, either Private or Public. As a result, it stores a single copy of each file and only the reference is passed to all the other transactions where the same file is under consideration.

The outcomes of the proposed system are suggesting that this strategy is very beneficial in harvesting the cloud storage by removing the duplicated data as shown in the Tab. 3 for Private CSPs & Tab. 4 for Public CSPs, given ahead:

Table 3: Harvesting the memory from private CSPs without using federation agent in federated cloud

S. No.	Name of the file	File size	User_id	Cloud_name	Action_status	Space after deduplication
1	File1	100	User1	CSP 1	Stored	100
2	File1	100	User2	CSP 1	Stored	100
3	File1	100	User1	CSP 1	Denied	–
4	File1	100	User1	CSP 2	&Reference	–
5	File2	200	User1	CSP 1	Stored	200

Table 4: Harvesting the memory from public CSPs without using federation agent in federated cloud

S. No.	Name of the file	File size	User_id	Cloud_name	Action_status	Memory after deduplication
1.	File11	200	–	CSP3	Stored	200
2.	File11	200	–	CSP3	&Reference	–
3.	File11	200	–	CSP4	&Reference	–
4.	File12	300	–	CSP3	Stored	300
5.	File1	100	–	CSP3	Stored	100
6.	File2	200	–	CSP3	Stored	200
7.	File1	100	–	CSP4	&Reference	–

As shown in the Tab. 3, before applying any deduplication strategy, the total memory needed to store the above data was 600 kbs. After applying the proposed deduplication policy, the storage requirement reduces to 400 kbs for the same data and 200 kbs of memory is saved.

As shown in the Fig. 4 a graphical (3-D Cluster-Column) representation of the simulation results for memory optimisation in private storage, the total memory consumption for the set of files used in the existing private storage was 600 kbs and for the same set of files with proposed deduplication strategy the memory requirement was 400 kbs. It empirically proves that the proposed system is harvesting 33.3% of memory from redundant storage in Private CSPs of Federated clouds.

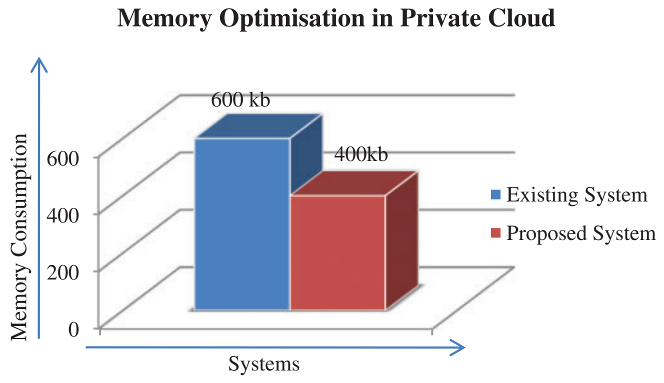


Figure 4: A graphical representation of the simulation results for memory optimization in private storage

As shown in the Tab. 4 for public CSPs, after applying the proposed strategy the memory requirement was reduced to 800 kbs which was otherwise 1300 kbs.

As shown in the Fig. 5 a graphical (3-D Cluster-Column) representation of the simulation results for memory optimization in public storage, the total memory consumption for the set of files used in the existing public storage was 1300 kbs and for the same set of files with proposed deduplication strategy the memory requirement was 800 kbs. It indicates that the proposed system is conserving 38.46% of memory from redundant storage in Public CSPs of Federated clouds.

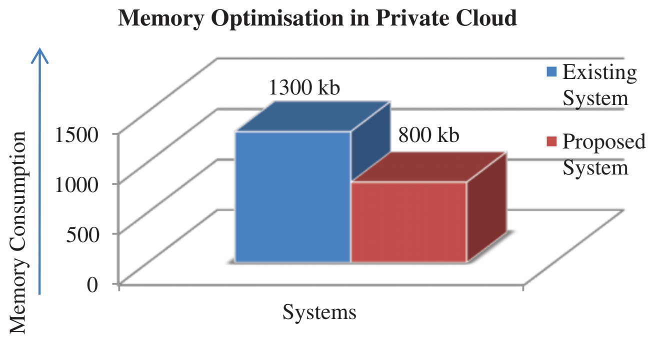


Figure 5: A graphical (3-D Cluster-Column) representation of the simulation results for memory optimization in public storage

As shown in the Tab. 5, after using Federation agent, it was observed that 1100 kbs of memory is saved which is 400 kbs more as compared to the system without Federation Agent in the Federated cloud environment. Moreover, as all the transactions are recorded in one federation table, lookup operation also takes lesser time than the system without federation agent.

Table 5: Harvesting the memory using federation agent in federated cloud

S. No.	Name of the file	File_size	User_id	Cloud_name	Action_status	Memory after Deduplication
1	File1	100	User1	CSP1	Stored	100
2	File1	100	User2	CSP1	&Reference	–
3	File1	100	User1	CSP1	Denied	–
4	File1	100	User1	CSP2	&Reference	–
5	File2	200	User1	CSP1	Stored	200
6	File11	200	–	CSP3	Stored	200
7	File11	200	–	CSP3	&Reference	–
8	File11	200	–	CSP4	&Reference	–
9	File12	300	–	CSP3	Stored	300
10	File1	100	–	CSP3	Stored	–
11	File2	200	–	CSP3	Stored	–
12	File1	100	–	CSP4	&Reference	–

As shown in the Fig. 6 a graphical (3-D Cluster-Column) representation of the simulation results for memory optimization in proposed system with federation agent, the total memory consumption for the set of files used in the existing private and public storage was 1900 kbs and for the same set of files with proposed deduplication strategy the memory requirement was 1200 kbs. After further optimization with the federation agent, the memory requirement for the same set of file decreased to 800 kbs which indicates that the proposed system with federation agent is conserving 57.89% of memory from redundant storage in federated cloud storage as compared to the existing system.

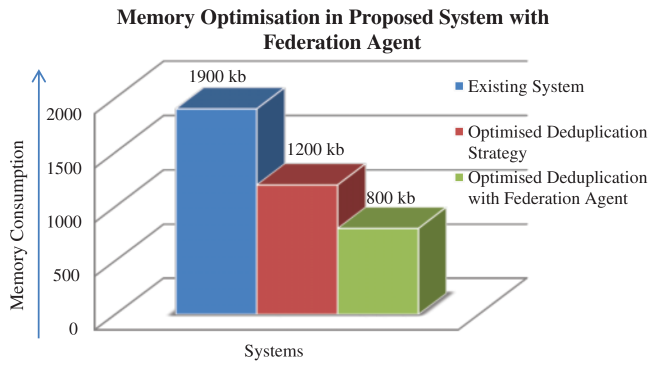


Figure 6: A graphical (3-D Cluster-Column) representation of the simulation results for memory optimization in proposed optimized deduplication strategy with federation agent

6 Conclusion

This paper presents the concept of Federation Agent to mollify the meagerness of the existing system and to further improve the data duplication strategy in cloud environment. Federation Agent is centrally managing all the transactions in common federation table and is well aware about the existence of any file in any of its participating CSPs, either Private or Public. The simulation of the proposed strategy has been performed on a cloud simulation tool, CloudSim 3.0.3, for implementing and testing the algorithm. The outcomes of the proposed system are suggesting that this strategy is very beneficial in harvesting the cloud storage by removing the duplicated data and conserving considerable amount of memory. It has been empirically proven that the system without Federation Agent is harvesting 33.3% memory from Private CSPs and 38.46% from Public CSPs than existing system, whereas, the proposed system with Federation Agent is conserving 57.84% cloud storage. In this study, file level deduplication technique has been used in the proposed system, in future, more research can be done on block level deduplication techniques in federated cloud environment. In addition to this, more alternatives can be sought for better indexing and faster lookup operations for deduplication in federated clouds.

Acknowledgement: This paper and research behind it would not have been possible without the exceptional support of my supervisor and research colleagues.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] V. Sharma and M. Bala, "An improved task allocation strategy in cloud using modified k-means clustering technique," *Egyptian Informatics Journal*, vol. 21, no. 4, pp. 201–208, 2020.
[2] G. Zangara, D. Terrana, P. P. Corso, M. Ughetti and G. Montalbano, "A cloud federation architecture," in *10th Int. Conf. on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, Krakow, Poland, pp. 498–503, 2015.
[3] V. Sharma and M. Bala, "A credits based scheduling algorithm with k-means clustering," in *First Int. Conf. on Secure Cyber Computing and Communication (ICSCCC)*, Jalandhar, Punjab, India, pp. 82–86, 2018.

[4] R. N. Calheiros, A. N. Toosi, C. Vecchiola and R. Buyya, “A coordinator for scaling elastic applications across multiple clouds,” *Future Generation Computer Systems*, vol. 28, no. 8, pp. 1350–1362, 2012.

[5] B. K. Rani, B. P. Rani and A. V. Babu, “Cloud computing and inter-clouds–types, topologies and research issues,” *Procedia Computer Science*, vol. 50, pp. 24–29, 2015.

[6] P. Sowmaya and R. Kumar, “An efficient approach to implement federated clouds,” *Asian Journal of Pharmaceutical and Clinical Research*, vol. 10, no. 13, pp. 40–44, 2017.

[7] T. Kurze, M. Klems, D. Bermbach, A. Lenk, S. Tai *et al.,* “Cloud federation,” in *2nd Int. Conf. on Cloud Computing GRIDs, and Virtualization*, Rome, Italy, pp. 32–38, 2011.

[8] D. G. Kogias, M. G. Xevgenis and C. Z. Patrikakis, “Cloud federation and the evolution of cloud computing,” *Computer*, vol. 49, no. 11, pp. 96–99, 2016.

[9] A. Kayed and O. Shareef, “Survey on federated clouds,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 2, pp. 83–92, 2015.

[10] N. Grozev and R. Buyya, “Inter-cloud architectures and application brokering: Taxonomy and survey,” *Software Practice and Experience*, vol. 44, no. 3, pp. 369–390, 2012.

[11] R. Buyya, R. Ranjan and R. N. Calheiros, “InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services,” *Lecture Notes in Computer Science*, vol. 6081, pp. 13–31, 2010.

[12] A. Kertesz, “Characterizing cloud federation approaches,” in *Cloud Computing, Computer Communications and Networks*, Springer, Cham, pp. 277–296, 2014. https://doi.org/10.1007/978-3-319-10530-7_12.

[13] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg and I. Brandic, “Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility,” *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[14] E. Birrell and F. B. Schneider, “Federated identity management systems: A privacy-based characterization,” *IEEE Security & Privacy*, vol. 11, no. 5, pp. 36–48, 2013.

[15] Z. Yan, L. Zhang, W. Ding and Q. Zheng, “Heterogeneous data storage management with deduplication in cloud computing,” *IEEE Transactions on Big Data*, vol. 5, no. 3, pp. 393–407, 2017.

[16] N. Chhabra and M. Bala, “A comparative study of data deduplication strategies,” in *First Int. Conf. on Secure Cyber Computing and Communication (ICSCCC)*, Jalandhar, Punjab, India, pp. 68–72, 2018.

[17] J. R. Douceur, A. Adya, W. J. Bolosky, P. Simon and M. Theimer, “Reclaiming space from duplicate files in a serverless distributed file system,” in *22nd Int. Conf. on Distributed Computing Systems*, Vienna, Austria, pp. 617–624, 2002.

[18] M. Bellare, S. Keelveedhi and T. Ristenpart, “Message-locked encryption and secure deduplication,” in *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, vol. 7881, pp. 296–312, 2013. https://doi.org/10.1007/978-3-642-38348-9_18.

[19] M. Bellare, S. Keelveedhi and T. Ristenpart, “DupLESS: Server aided encryption for deduplicated storage,” in *22nd USENIX Conf. on Security*, Washington, D.C, pp. 179–194, 2013.

[20] J. Li, X. Chen, M. Li, J. Li, P. P. C. Lee *et al.,* “Secure deduplication with efficient and reliable convergent key management,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 6, pp. 1615–1625, 2014.

[21] J. Liu, N. Asokan and B. Pinkas, “Secure deduplication of encrypted data without additional independent servers,” in *22nd ACM SIGSAC Conf. on Computer and Communications Security-CCS ’15*, Denver, Colorado, USA, pp. 874–885, 2015.

[22] M. L. Dhore, K. M. Varpe and R. M. Dhore, “Secure deduplication of encrypted data in cloud,” *International Journal of Future Generation Communication and Networking*, vol. 13, no. 1, pp. 1594–1600, 2020.

[23] C. Fan, S. Huang and W. Hsu, “Hybrid data deduplication in cloud environment,” in *Int. Conf. on Information Security and Intelligent Control*, Yunlin, Taiwan, pp. 174–177, 2012.

[24] J. Li, X. Chen, X. Huang, S. Tang, Y. Xiang *et al.,* “Secure distributed deduplication systems with improved reliability,” *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3569–3579, 2015.

[25] J. Wu, Y. Li, T. Wang and Y. Ding, "A confidentiality-preserving deduplication cloud storage with public cloud auditing," *IEEE Access*, vol. 7, pp. 160482–160497, 2019.
[26] X. Liang, Z. Yan and R. H. Deng, "Game theoretical study on client-controlled cloud data deduplication," *Computers & Security*, vol. 91, no. 2, pp. 101730, 2020.
[27] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.
[28] R. Kaur, I. Chana and J. Bhattacharya, "Data deduplication techniques for efficient cloud storage management: A systematic review," *The Journal of Supercomputing*, vol. 74, no. 5, pp. 2035–2085, 2018.
[29] N. Chhabra and M. Bala, "An optimized data duplication strategy for federated clouds using bloom filters," *International Journal of Future Generation Communication and Networking*, vol. 13, no. 4, pp. 2684–2693, 2020.