Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 6, Issue. 5, May 2017, pg.238 – 249

Secure Key Distribution and Deduplication of Data

Prof. Dolly Chandani[1], Pooja Kumbhar[2], Bhagyashree Mahajan[3], Poonam Dambare[4], Pratiksha Ladkat[5]

Professor Computer Engineering, KJCOEMR, Pune
Student Computer Engineering, KJCOEMR, Pune
Student Computer Engineering, KJCOEMR, Pune
Student Computer Engineering, KJCOEMR, Pune
Student Computer Engineering, KJCOEMR, Pune

[1] poojakumbhar6896@gmail.com; [2] mahajanrinku@ymail.com; [3] dambarepoonam1212@gmail.com; [4] pratikshaladkat30@gmail.com

Abstract— In this paper, we are using hash code for the content of the file, if this code is find into the database then system gives only reference of file to the another users whose uploading same file and file will be divide into three chunks which is stored on the three different location so the load will be divide and automatically load balancing happened.

Keywords— Secure distribution, Data deduplication, Replica

I. INTRODUCTION

Cloud storage systems are able to provide low-cost and convenient network storage service for users, which makes them more and more popular. However, the storage pressure on cloud storage system caused by the explosive growth of data is growing by the day, especially a vast amount of redundant data waste plenty of storage space. Data deduplication can effectively reduce the size of data by eliminating redundant data in storage systems. However, current researches on data deduplication, which mainly focus on the static scenes such as the backup and archive systems, are not suitable for cloud storage system due to the dynamic nature of data. In the storage nodes (Snodes), DelayDedupe, a delayed target- deduplication scheme based on the chunk-level deduplication and the access frequency of chunks, are proposed to reduce the response time. Combined with replica management, this method determines whether new duplicated chunks for data modification are hot and removes the hot duplicated chucks when they arent hot. The experiment results demonstrate that the DelayDedupe mechanism can effectively reduce the response time and achieve the storage load of Snode more balanced.

II. RELATED WORK

A. OBJECTIVES

- Detection and elimination of duplicated data
- Avoid the duplicated data

B. PROBLEM DEFINITION

The problem is to determine how to design secure deduplication systems with higher reliability in cloud computing. Hence it is been proposed in the distributed cloud storage servers into deduplication systems to provide better fault tolerance. To protect data confidentiality, the secret sharing technique is utilized, which is also compatible with the distributed storage systems. To support deduplication, a short cryptographic hash value of the content will also be computed and sent to each storage server as the fingerprint of the fragment stored at each server.

C. LITERATURE SURVEY

- In the current cloud server storage techniques there is a less security for the Update, Delete and Download file.
- There is less load balancing techniques and no Deduplication. The current system has only provides the spaces on the server but not avoid the duplicate files.
- In this project we are using Secure Hash Algorithm for the content of the file.
    - Duplicate file message.
    - Generates Three chunks
    - Chunks stored at different locations.

1. MOTIVATIONS

- Handling huge data on cloud is messy.
- It is difficult to upload as well as download mass data
- Hence there should be centralized system for cloud.

2. EXISTING SYSTEM

Recent years have witnessed the popularity of cloud computing, mobile computing and the Internet of things which brings about the explosive growth of data. To meet users' demands for low-cost and convenient storage services, cloud storage has already been a typical storage system.

III.PROPOSED SYSTEM

In this system we check the file for avoid duplication with content of file using the SHA algorithm and generate the hash code for each file and if it is same then only the reference of the file is given to the another user and while uploading the file, it is divided into three different chunks and load balancing is achieved.

PROPOSED MODEL

In this paper, we propose the architecture of deduplication system for cloud storage environment and give the process of avoiding duplication at the file-level and chunk-level on the client side.

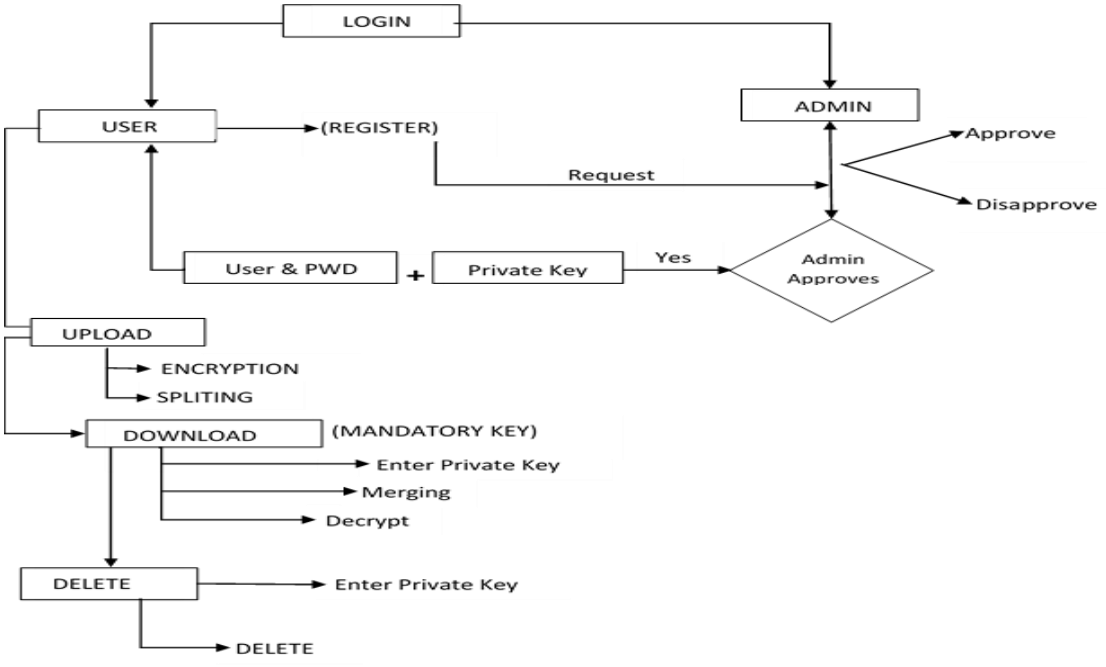


Fig 1:- Proposed System

The following figure first user register for his account and then the request is goes to the admin. Admin checks the information which is in registration form then he decides to approve or disapprove, if admin approved the users request then the mail of password and private key is send to this particular user and if admin disapproved his request then mail of disapproved request is send to this user.

User login with email id and password and then he can perform the upload, download and view operations with his private key which is provided by the admin.

The private key is necessary while performing operations on the cloud data.

When we are uploading the file then the file will divide into three different chunks and save to the cloud storage and then the file has uploaded and when we downloading this uploaded file then at this time the file gets merge in our original format.

IV. ALGORITHMS

AES Algorithm

Step 1:

Key Expansion Using Rijndael's key schedule Round keys are derived from the cipher key.

Step 2:

If Distance To Tree(u) > Distance To Tree(DCM) and First Sending then

Step 3:
Initial Round

Add Round Key where Each byte of the state is combined with the round key using bitwise XOR

Step 4: Round

- Sub Bytes : non-linear substitution step
- Shift Rows : transposition step
- Mix Columns : mixing operation of each column.
- Add Round Key

Step 5: Final Round: It contain Sub Bytes, Shift Rows and Add Round Key

SHA-1 Algorithm

Step 1:
attach Padding Bits.... Message is "padded" with a 1 and as many 0's as important to bring the message length to 64 bits lesser than an even multiple of 512.

Step 2:
Attach Length....64 bits are added to the end of the padded message. These bits hold the binary format of 64 bits showing the length of the original message.

Step 3:
Prepare Processing Functions.... SHA1 requires 80 processing functions defined as:
f(t;B,C,D) = (B AND C) OR ((NOT B) AND D) (0 <= t <= 19)
f(t;B,C,D) = B XOR C XOR D (20 <= t <= 39)
f(t;B,C,D) = (B AND C) OR (B AND D) OR (C AND D) (40 <= t <=59)
f(t;B,C,D) = B XOR C XOR D (60 <= t <= 79)

Step 4:
Create Processing Constants....SHA1 needs 80 processing constant words defined as:
K(t) = 0x5A827999 (0<= t <= 19) K(t) = 0x6ED9EBA1 (20 <= t <= 39) K(t) 0x8F1BBCDC (40 <= t<=59)
K(t) = 0xCA62C1D6 (60 <= t <= 79)

Step 5:
Initialize Buffers.... SHA1 needs 160 bits or 5 buffers of words (32 bits) H0 = 0x67452301
H1 = 0xEFCDAB89
H2 = 0x98BADCFE
H3 = 0x10325476
H4 = 0xC3D2E1F0

Step 6:
Processing Message in 512 bit blocks (L blocks in total message)....
This is the an important task of SHA1 algorithm which loops through the padded and added message
in 512 bit blocks. Input and predefined functions:
M[1, 2, ..., L]: Blocks of the padded and added message

f(0;B,C,D), f(1,B,C,D), ..., f(79,B,C,D): 80 Processing Functions
K(0), K(1), ..., K(79): 80 Processing Constant Words H0, H1, H2, H3, H4,H5: 5 Word buffers with starting values

V. MATHEMATICAL MODEL

Set Theory Analysis:

a. Let 'S' be the | Load Balancing in Cloud as the final set

S = {……

b. Identify the inputs as D, Z, N, F

S = {D, Z, N, R, Q …

D = {D1, D2, D3…| 'D' gives Data to be stored or download from cloud}
Z = { Z1, Z2, Z3 …| 'Z' is t he size of data}
N = { N1, N2, N3 …| 'N' is Number of clouds }
R = { R1, R2, R3 …| 'R' is fragmented data }
Q = { Q1, Q2, Q3 …| 'Q' is request to download data }

c. Identify the outputs as O

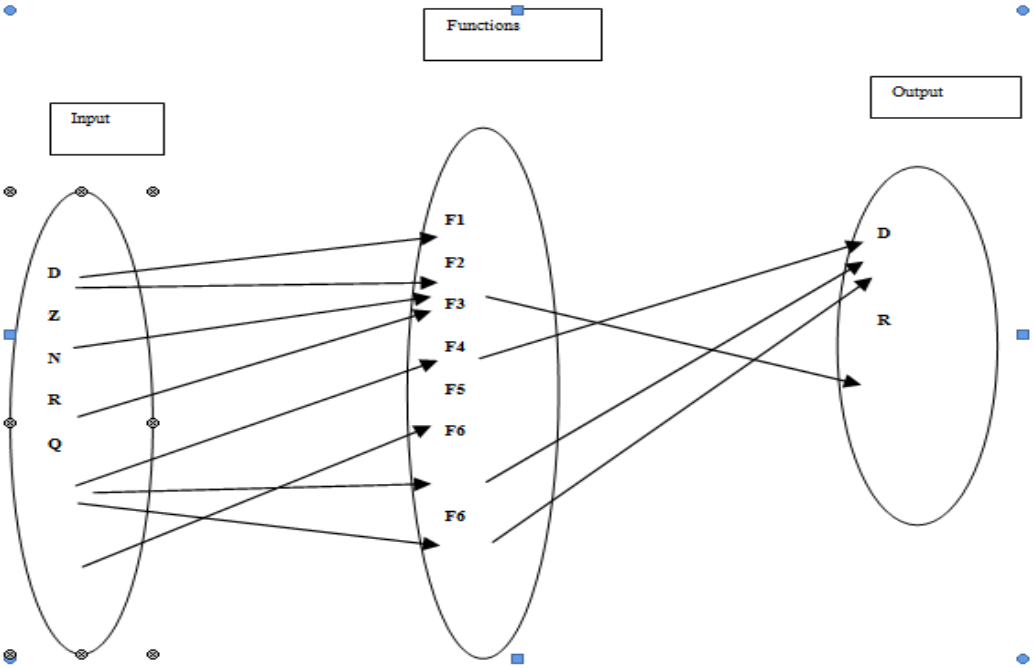
S = {D, Z, N, R, Q…
D = {D1, D2, D3…| 'D' gives Data to be stored or download from cloud}
R = { R1, R2, R3 …| 'R' is fragmented data }

d. Identify the functions as 'F'

S = {D, Z, N, R,Q, F…
F = {F1(), F2(), F3(), F4(), F5(), F6()}

F1(D) :: Upload data
F2 (D, Z, N) :: divide data into equal fragments compared to no of clouds
F3 (R) :: store data
F4 (Q) :: Request for download
F5 (R) :: Combine fragments of data
F6 (R) :: Download

Hence the functionality can be shown as,



ARCHITECTURE

A. In this architecture, client uploading the file then the file is getting divides into the different chunks and stored with different location. The operation is performing through the centralized coordinator which contain three blocks first is duplication engine, second is walrus controller and third one is load balancer this blocks are used for load balancing.

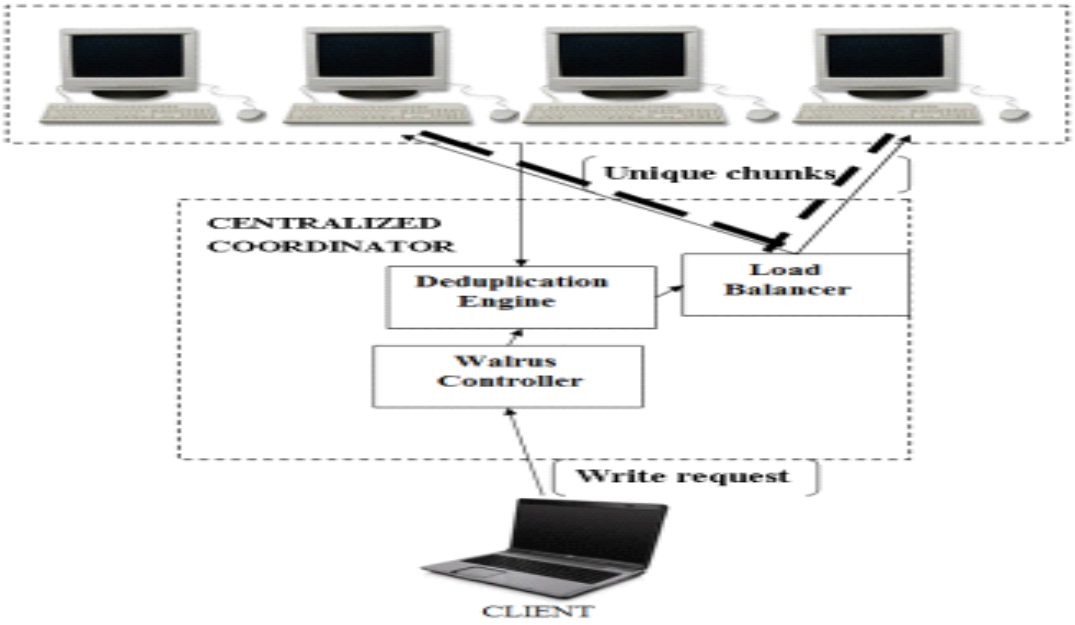


Fig 2:- Architecture-1

B. As depicted in Fig. 3, the system consists of Client, MS, Secondary MS (SMS) and Snode. The terms ”Client” and ”Snode” represent the location of the original data to be uploaded and the location of the new data to be stored after deduplication, respectively. We suppose that there are m Clients and n Snodes. Users send the request of file uploading, access, modification, and deletion through Client. All the metadata of files are stored in MS and the actual data in Snode. With the metadata information, we can find the location of data in Snode and determine whether the data from Client is duplicated. MS plays the role as the core manager of the whole system architecture. Once MS malfunctions, the system will break down. To avoid single-point failure, SMS is responsible for synchronizing the backup of metadata images and operation logs. Every time we upload some data from Client to Snode, we first deal with these data by local deduplication and upload their metadata information to MS. Then we wait for the answer about non-duplicated data from MS, and finally upload the new data to Snode. There are four modules on the Client side: File Preprocess Module, Local Deduplication, Metadata Manager, and File Transfer Module

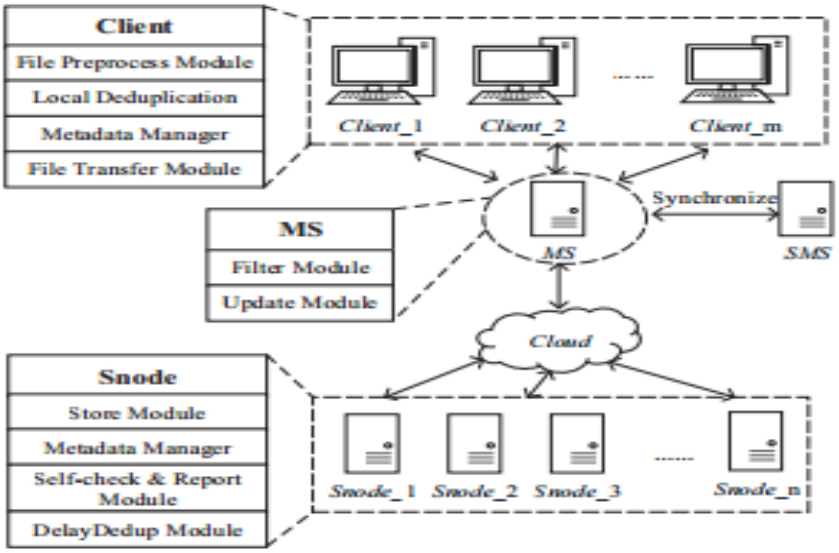


Fig 3:- Architecture-2

There are four modules on the Snode side: Store Module, Metadata Manager, Self-check & Report Module, and DelayDedupe Module. Store Module is used to store the actual data blocks on the disk.

SOFTWARE REQUIREMENT

Memory requirements:

500 GB of Hard disk space
4 GB RAM
Specific technologies, tools :
Java (Eclipse Kepler)
Apache Tomcat Server7 JDK 1.7

Databases:

MySQL Workbench 5.0
Language requirements :
Java

HARDWARE REQUIREMENT

CPU type: Intel Pentium 4
Clock speed: 3.0 GHz
Ram size: 4 GB
Hard disk capacity: 500 GB
Keyboard type: Internet keyboard

ACTIVITY DIAGRAM FOR SYSTEM

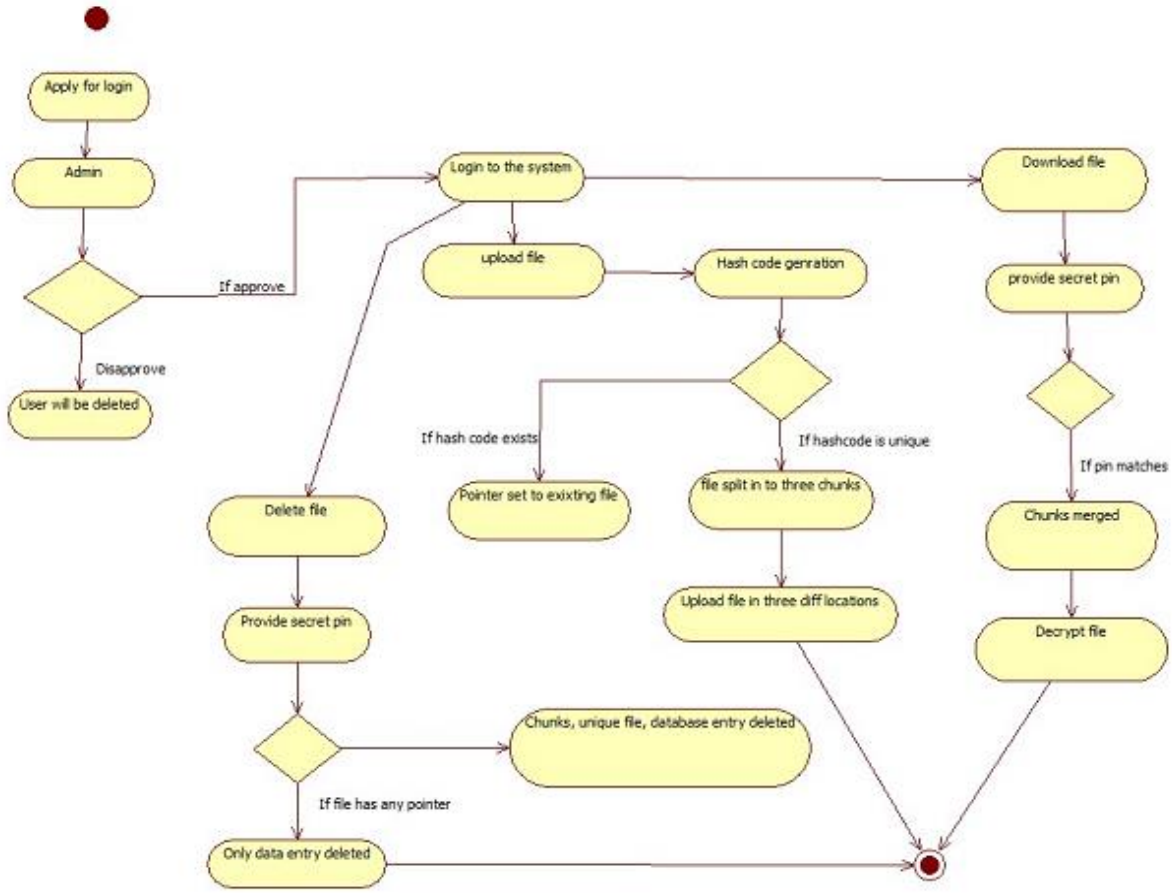


Fig 4:- Activity diagram

RESULTS AND DISCUSSION

The Random Number Generation Algorithm has been applied in the system successfully. One of the experimental results of the Random Number Generation Algorithm is that it makes two shares of data after data uploading or downloading process. During login phase, after image validation, OTP (One Time Password) is generated and verified.

The following fig. shows that the entry in database which the user logged in.

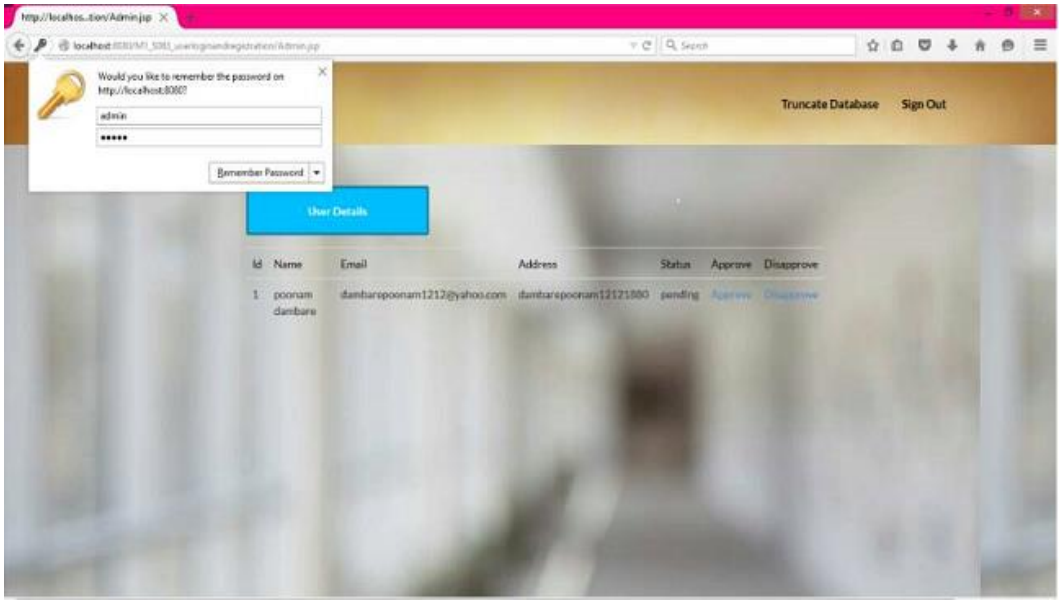


Fig 1:- Database Entry

The following fig shows that the admin approved the user request to upload data in cloud and the security key provided by the admin to users mail id

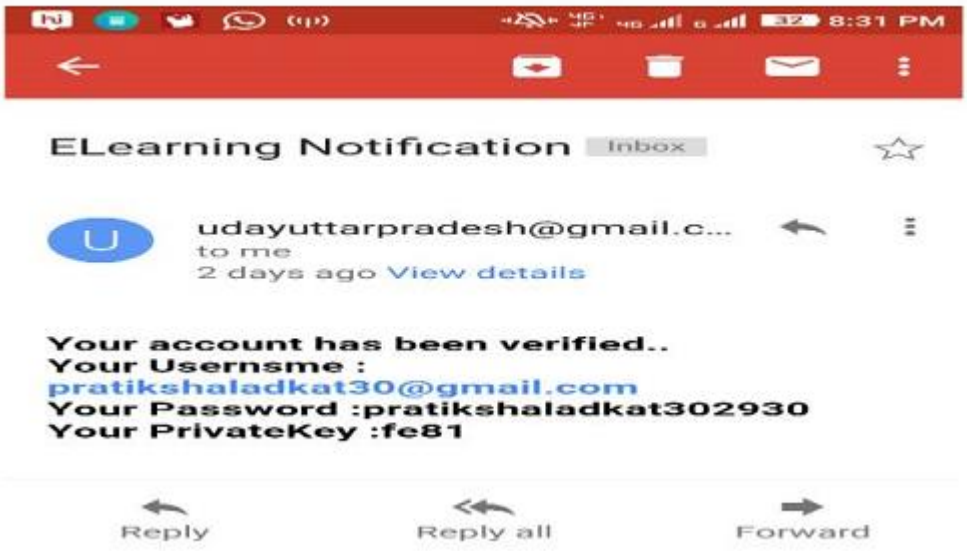


Fig 2:- Security key provided by admin

The fig shows that to enter security key for upload data

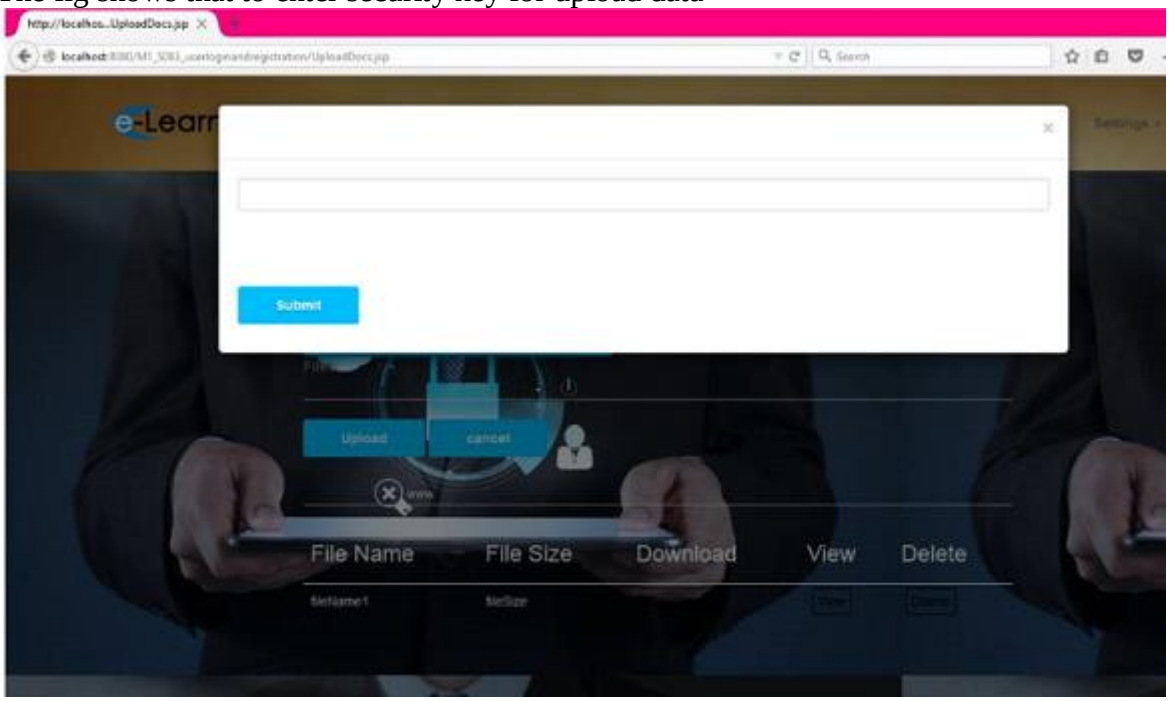


Fig 3:- Security key

This fig shows that upload the data to cloud.

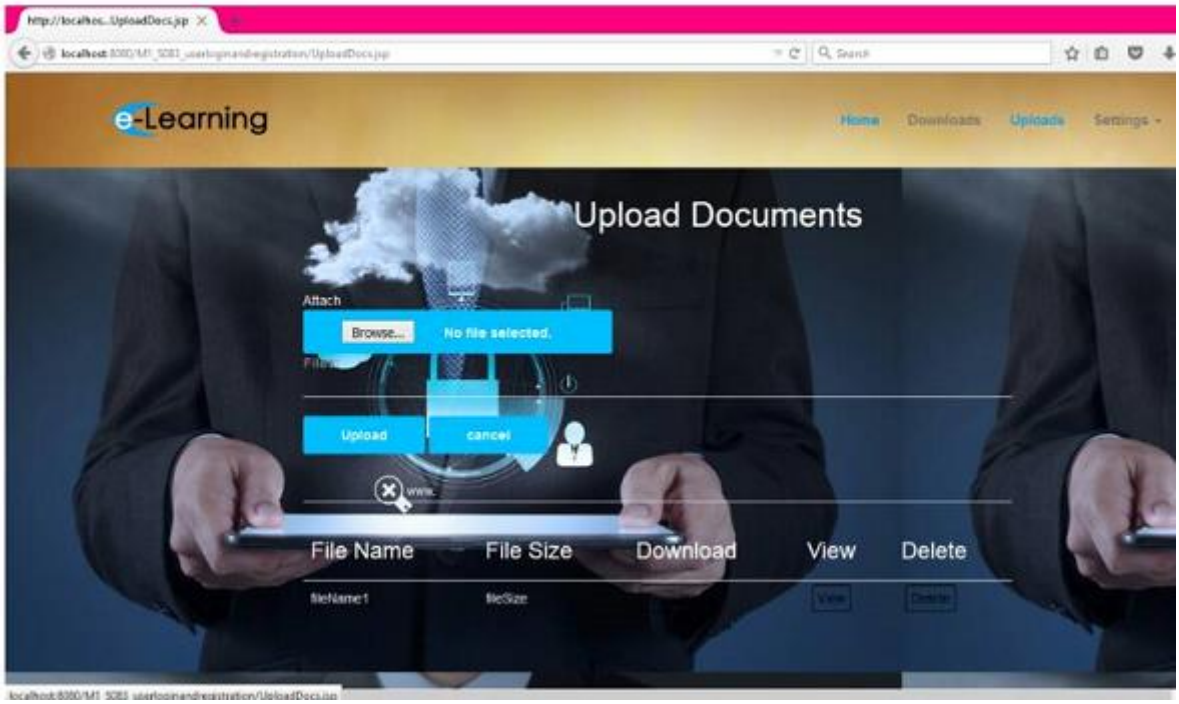


Fig 4:- Upload file

This fig shows that to download the files from cloud

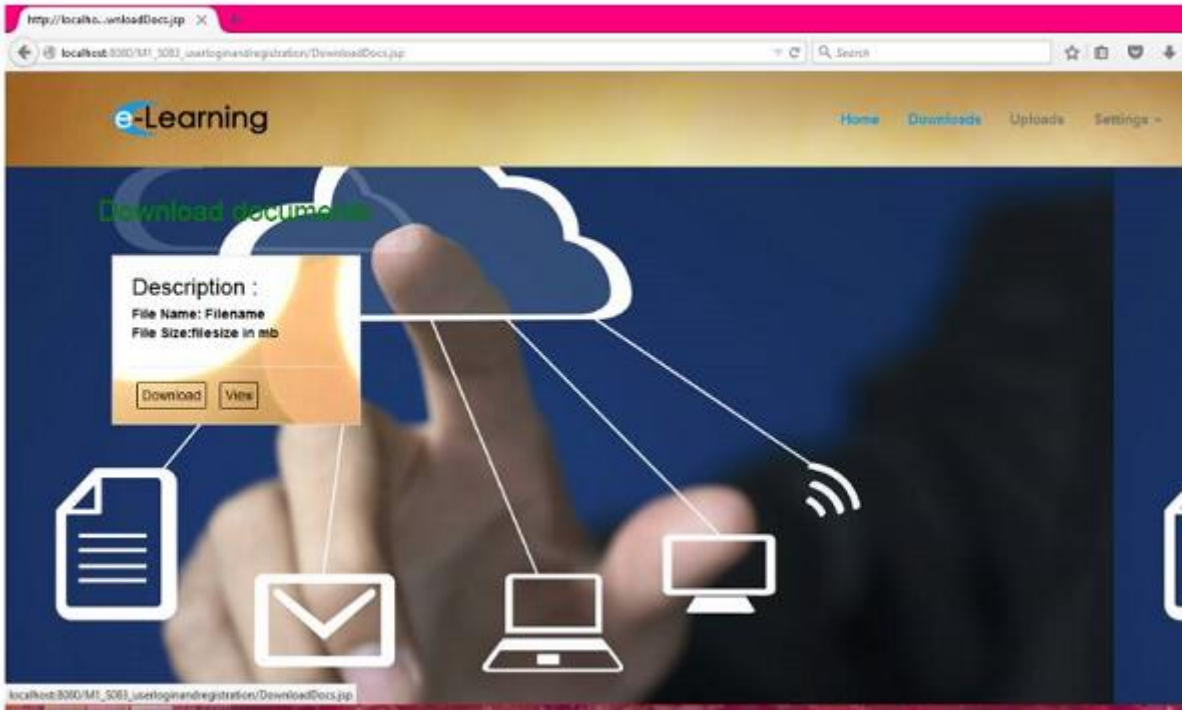


Fig 5:- Download file

VI. CONCLUSION

In this paper, we propose an architecture of the deduplication system for cloud storage environment and give a process of avoiding duplication in the each stage. In Client, we employ a file-level and chunk-level deduplication to avoid duplication. Meanwhile, we propose the DelayDedupe strategy in Snode, a target-deduplication scheme based the chunk-level deduplication and the access frequency of chunks to eliminate redundancy. Each Snode determines whether the new duplicated chunks caused by the data modification are hot and remove a non-hot duplicated chunks in those Snodes a storage load of which is relatively smallest. Experiment results show that DelayDeduplication can indeed reduce the response time of system and achieve the storage load more balanced. However, it is the possible for system to misjudge hot chunk because we only choose two time slices before the current time to evaluate the tendency of data access. We plan to improve the accuracy of predicting a hot duplicated chunk to optimize the response time as our future work. And using divide chunks method we achieve the load balancing.

ACKNOWLEDGEMENTS

We would like to thank the reviewers for their detailed comments and suggestions throughout the reviewing process that helped us significantly improve the quality of this paper. This work was jointly sponsored by the National Natural Science Foundation of China under Grants 61472192, 61202004 and the Special Fund for Fast Sharing of Science Paper in Net Era by CSTD under Grant 2013116.

REFERENCES

[1] J. Wu, L. Ping, X. Ge, Y. Wang, and J. Fu, ”Cloud storage as the infrastructure of cloud computing,” in Proc. 2010 Int. Conf. Intell. Comput. Cognitive Inform. (ICICCI), Kuala Lumpur, 2010, pp. 380-383.
[2] J. Gantz and D. Reinsel, ”The digital universe decade-Are you ready,” IDC White Paper, http://www.emc.com/collateral/analyst-reports/idc-digitaluniverse-are-you-ready.pdf, 2010.
[3] P. Xie, ”Survey on deduplication techniques for storage systems,” Comput. Sci., vol. 41, no. 1, pp. 22-30, Jan. 2014.
[4] H. Biggar, ”Experiencing data de-duplication: Improving effi- ciency and reducing capacity requirements,”ESG White Paper, http://www.dcs-os.it/files/ESG De-Dupe Feb07.pdf, 2007.
[5] http://esatjournals.net/ijret/2013v02/i06/IJRET20130206011.pdf
[6] http://www.ijcert.org/V2I1281.pdf

