Block Level De-duplication Check for Shared Data on Hybrid Cloud using Convergent Key

Priyanka N. Patil
Department of Computer Engineering,
GES's R. H. Sapat College of Engineering,
Savitribai Phule Pune University, India

C. R. Barde
Department of Computer Engineering,
GES's R. H. Sapat College of Engineering,
Savitribai Phule Pune University, Indianilesh.

ABSTRACT
In regards to increase in use of digital information users prefer to store information in cloud system. In cloud storage system many users can store same type of data leading to data duplication causing a high utilization of bandwidth. Some techniques are proposed for making cloud more efficient and effective regarding to storage and bandwidth. In current time data de-duplication is effective technique to avoid such data duplication caused due to privileged as well as non-privileged user. To save bandwidth to transact data when replicating it offsite for disaster recovery huge organization, companies and all education institutes supports de-duplication technique. With the help of "Content hash keying" data confidentiality is provided. Using this technique data is first encrypted and the encrypted data is outsourced to the client. To address the problem of authorized access in our proposed system de-duplication check technique is introduced and privileged access which is different from traditional data de-duplication check system. Log based approach for unauthorized data duplication check in hybrid cloud architecture is also explored. For privileged as well as for non-privileged users de-duplication can be managed using the above technique. Proposed system provides clever solution for duplication of data and also works on bandwidth efficiency.

Keywords
Cloud, Content Hash Keying, Convergent Key De-Duplication, Encryption

1. INTRODUCTION
This project focuses on a method to manage data redundancy on cloud caused due to massive data transaction by cloud users. De-duplication technique avoids storage of repetitive data on cloud. This makes cloud more responsive by saving storage space and its bandwidth. Duplicate replica of text file is traced and eliminated. Along with this de-duplication technique the project covers security aspect on cloud. Users encrypt their data and save cipher texts on cloud. This encrypted data creates some difficulty to compare same data. If two users try to upload same file then it will be encrypted and separate cipher texts are stored for two users on cloud. Hence different cipher texts for same file are stored leading to data redundancy. Convergent encryption technique is used for this bottle neck problem. In this technique de-duplication is possible and secrecy of data is preserved by encryption. In Convergent key technique / Content Hash Keying process, same cipher text is generated for same file though keys are different. Same cipher text for same file is helpful to compare and provide de-duplicate facility.

For effective bandwidth constraints on uploading and down-loading of files which consume considerable bandwidth is put. Also uploading of file which is already present can be avoided. Regarding the presence of same kind of file, metadata of file generated during uploading is matched with metadata of already existing file. Such tag matching or Meta data matching technique improves the usage of bandwidth.

Problem Definition
Knowingly or unknowingly multiple copies of data can be uploaded to the cloud storage due to increase of cloud utilization for data storage as well as sharing. Due to duplicated data on public cloud, its storage efficiency and bandwidth efficiency is not utilized properly. De-duplication is one of the important compression techniques.

This technique removes duplicate copies of data and reserve a single copy. In this technique, upload data is compared with the data on public cloud and such duplication is avoided. Data is encrypted by the user for the security reason using his/her private key. As key changes various copies of same data will be generated. So the main problem is the comparison of encrypted data on cloud same file.

2. LITERATURE REVIEW
Cloud computing is now an emerging market. Day by day application hosting on cloud increases rapidly causes huge data storage on cloud. Due to this the main challenge faced by cloud service provider is the management of this ever increasing bulk data.

S. Quinlan and S. Dorward. Venti[2] In 2002, write-once policy of data approach is used. It provides efficient storage applications such as backup system.

On the similar lines, P. Anderson and L. Zhang [3] proposed a system for laptop and mobile backup system. The backup is in encrypted format. This paper mainly focuses on increasing the speed of backup, and reducing the storage requirements.

The above mentioned paper only focuses on data backup de-duplication technique. Using above technique only the user specific data de-duplication is avoided but not the complete data de-duplication.

J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer [4] introduced convergent key technique. Which resulted in increased data confidentiality and feasible data de-duplication).By applying cryptographic hash function on data convergent key is generated. Using this key it encrypts/decrypts a data. Encrypted data is sent to the cloud and user preserves the key and sends the cipher text to the cloud.

The encryption is deterministic operation. The key is derived from the data content, hence identical data copies will generate the same convergent key and using the same key same cipher text is generated.

Pinkas, and A. Shulman-Peleg [5] secure proof of ownership protocol is also needed to prevent unauthorised access. When same data is found this proof is used. After the proof, subsequent users with the same file will be provided a pointer from the server without needing to upload the same file. A user can download the encrypted file with the pointer from the server, which can only be decrypted by the corresponding data owners with their convergent keys.

Some Security proofs systems [10] provide either security proofs or attacks for a large number of identity-based identification and signature schemes defined either explicitly or implicitly.

M. Bellare and A. Palacio. Gq and schnorr [9] introduced first security proofs for these schemes under assumptions related to the underlying one-way functions. Both results extend to establish security against impersonation under concurrent attack. The formal security definition for PoW, roughly follows the threat model in a content distribution network, where an attacker does not know the entire file, but has accomplices who have the file. The accomplices follow the "bounded retrieval model", such that they can help the attacker obtain the file, subject to the constraint that they must send fewer bits than the initial min-entropy of the file to the attacker.

M. Bellare, S. Keelveedhi, and T. Ristenpart[6] Message-locked encryption and secure de-duplication: Message-Locked Encryption (MLE), where the key under which encryption and decryption are performed is itself derived from the message. MLE provides a way to achieve secure de-duplication.

Weak leakage-resilient client-side de-duplication of encrypted data in cloud storage by Xu et al. [13] introduced convergent encryption for efficient encryption.

The proposed technique only focuses on encryption and file level de-duplication not block level.

D. Ferraiolo and R. Kuhn. [8] Role-based access controls: In this they represent limitation of Mandatory Access Controls (MAC) technique. This is required for high level security like multilevel secure military applications. Discretionary Access Controls (DAC) focusing on security processing needs of industry and civilian government. This paper enlists the limitations of DAC as the principal access control method. This method is unfounded and inappropriate for personalized civilian access.

Architecture for secure cloud computing - Bugiel et al. [11] It provided an architecture consisting of twin clouds for securely outsourcing of user private data and arbitrary computations to an untrusted commodity cloud.

Privacy aware data intensive computing on hybrid clouds - Zhang et [12] al also presented the hybrid cloud techniques to support privacy-aware data-intensive computing.

We consider addressing the authorized privileged de-duplication problem over data in public cloud. The security model of our systems is similar to those related work, where the private cloud is assume to be honest but curious.

S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-[14] Proposes of POW (proof of ownership) technique is that a user can efficiently prove to the cloud storage server that he/she owns a file without uploading the file itself. It also proposes the Merkle-Hash Tree to enable client-side de-duplication, which include the bounded leakage setting.

The proposed scheme is focusing only on the data ownership and not on the data privacy.

In S. Ossowski and P. Lecca [15] extended proofs of ownership mechanism for encrypted files. These papers do not address how to minimize the key management overhead.

Pietro and Sorniotti [16] proposed efficient PoW scheme by choosing the projection of a file onto some randomly selected bit-positions as the file proof But this project do not deal with data privacy.

3. PROPOSED WORK

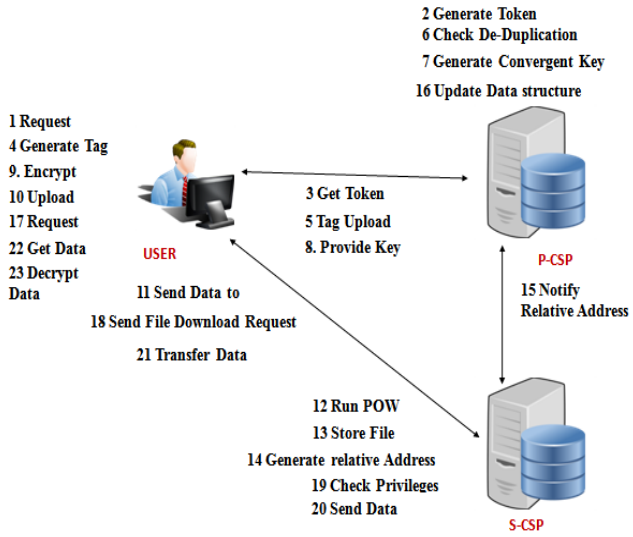

Fig.1 Architecture of proposed system

The above Fig.1 shows the basic structure of proposed system. User, S-CSP and P-CSP are different sections of system. P-CSP is secured cloud storage provider that authenticates as well as provides convergent key to the user. A user generates file tag and uploads to the S-CSP to check de-duplication. Increase in efficiency and reduction in bandwidth usage is achieved by uploading tag instead of complete file. De-duplication at block level is checked. If de-duplication found proof of ownership is executed. S-CSP notify to the user if some blocks or no block are found at S-CSP. The file is encrypted using convergent key and uploaded to the Storage CSP by the user.

Our proposed work mainly focuses on outsourcing the key management to the P-CSP, block level de-duplication and efficient bandwidth usage is the mainly focused in the project. If a user with privilege access demands a file for download from S-CSP it receives a convergent key further used for decryption.

3.1 Methodology

This system is divided in to two sections one is upload file and another is download file

Methodology for File Upload:

1. Register user data on P-CSP
2. User Login on P-CSP
3. P-CSP return the identification token T to the user
4. User selects the file to upload and add the group members with whom the file will get shared.
5. File tag generation at user end using SHA-1
6. Send tags to P-CSP using HTTP connection
7. P-CSP checks privileges of user

8. If access Privilege Check passes then system will allow de-duplication check else gives error message
9. In de-duplication check, it matches the file tag with existing file tags

Case 1: If tag matches run proof of ownership and share link with other users
Case 2: If no file tag matches then it will check de-duplication at block level

Case I: If Partial Duplication found: It runs proof of ownership for partial no of blocks
-For new blocks generate convergent key and return token +key + block matching file information to the user
-User encrypts unmatched block data using convergent key
-Upload token +encrypted data and file info to S-CSP.
-S-CSP save the file block
-Generate relative address mapping of file block of a file
-Returns the token and relative address tag information to P-CSP
-P-CSP saves the token
-Run proof of ownership
-Share link with other users
Case II: If no duplication found generate convergent key and return token +key
-User encrypt file block data using convergent key
-Upload token +encrypted data to S-CSP
-S-CSP save the file block
-Generate relative address mapping of file block of a file
-Returns the token and relative address tag information to P-CSP
-P-CSP saves the token
-Run proof of ownership
-Share link with other users

Methodology for File Download:
1. Register user data on P-CSP
2. User Login on P-CSP
3. P-CSP return the identification token T to the user
4. User Ask for file to download to P-CSP
5. P-CSP checks the privileges of user.
6. If user has privileges it returns file info + decryption key to the user
7. User sends file info and token to S-CSP
8. S-CSP verifies the token and return file blocks to the user
9. User decrypts the block and generates the original file

3.2 Mathematical Model

S = {U, P-CSP, S-CSP}
U = {IU, OU, FU}
IU= {I1, I2, I3, I4}
I1 = user registration details
I2 = User login details
I3 = File to upload
I4 = File name to download

FU= {F1, F2, F3, F4, F5, F6, F7}
F1 = User registration Request
F2 = User Login Request
F3 =File selection and block generation
F4 = tag generation for file level and block level using sha-1 algorithm
F5 = File encryption using AES
F6 = File Decryption using AES
F7 = generate tag for user access

OU= {O1, O2, O3, O4}
O1 = File level Tag
O2= Bock level tag
O3 = Access Privilege Tag
O4 = Cipher text

P-CSP=IP, OP, FP
IP= {I1, I2, I3, I4, I5}
I1 = User Registration Data
I2 = User Login Data
I3=File Tags for matching
I4 = Access Privileges
I5 = File location relative address detail array

FP= {F1, F2, F3, F4, F5}
F1= User Registration
F2 = User Identity Check
F3 = De-duplication check using tag matching
F4 = Proof Of ownership
F5 = File information storage

OP= {O1, O2, O3, O4}
O1 = User authentication response
O2 = de-duplication response
O3 = data sharing link
O4 = Access Privilege token

S-CSP= {IS, OS, FS}
IS= {I1, I2, I3}
I1 = Encrypted file block
I2 = File location relative address detail array
I3 = File Access Privilege Token

FS = {F1, F2, F3}
F1 = File block Storage
F2 = Maintain file storage data structure
F3 = File download

OS = {O1, O2}
O2 = File location relative address detail array
O1 = File to Download

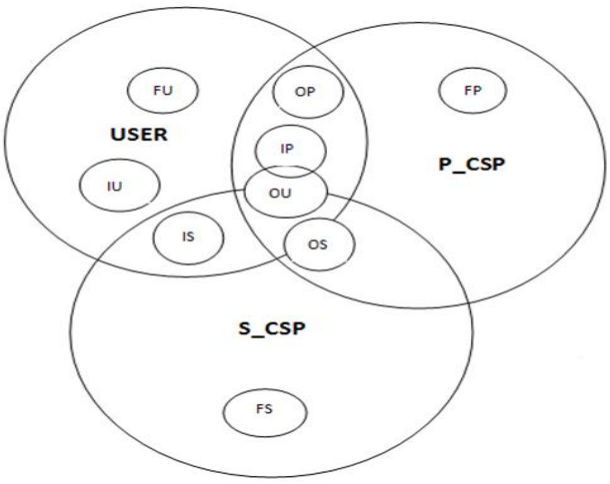

Fig.2 Design using set theory

Following fig.3 shows the state representation of our system.

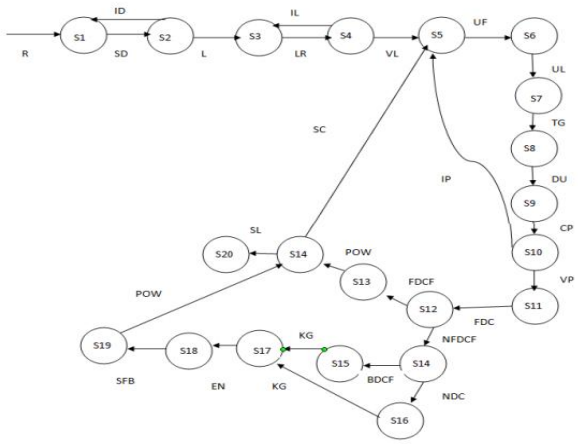

Fig.3 Mathematical module

Where,

R = Registration ID = Invalid Data SD = Save data L = Login IL = Invalid Login LR = Login request VL = Valid Login UF = Upload file UL = User list TG = Tag generation DU = Data upload CP = Check Privileges VP = Valid Privileges FDC = File de-duplication check	IP = Invalid Privileges NFDCF = No File Duplication found FDCF= File Duplication found BDCF=Block duplication found NDC = no duplication found KG = Key Generation EN =Encryption SFB = save file block POW=Proof of ownership SL = share link

3.3 Algorithms

In this section we discuss the algorithm of the models which will be used in the system.

1. AES Encryption:

Input: Plain text message m in Byte [] , Key k
Output: Cipher text message in byte []
Processing:
1. Define 4 * 4 state array
2. Define constant Nr = 4, R=16
3. Copy m in state[]
4. Add each byte of state[] to key k using ⊕
5. For Nr-1 rounds
Replace every byte in state[] with new value using lookup table
Shift last 3 rows of state[] upside cyclically
combine last 4 columns of state[]
Add each byte of state[] to key k using ⊕
end For
6. Shift last 3 rows of state[] upside cyclically
7. Add each byte of state[] to key k using ⊕
8. copy State[] to output[]

2. AES Decryption:

Input: Cipher text message C in byte[], Key k
Output: Plain text message m in Byte[]
Processing:
1. Define 4 * 4 state array
2. Define constant Nr = 4, R=16 ,
3. Copy C in state[]
4. Add each byte of state[] to key k using ⊕
5. For Nr-1 rounds
Inverse Replace every byte in state[] with new value using lookup table
Inverse Shift last 3 rows of state[] downside cyclically
combine last 4 columns of state[]
Add each byte of state[] to key k using ⊕
end For
6. Inverse Shift last 3 rows of state[] down word cyclically
7. Inverse Add each byte of state[] to key k using ⊕
8. copy State[] to output[]

3. SHA1:

Input: key, message m
Output: Hash Value hh
Processing:
1**. ml** = message length in bits
Initialize **h0** = 0x67452301,**h1** = 0xEFCDAB89,**h2** = 0x98BADCFE,**h3** = 0x10325476,**h4** = 0xC3D2E1F0
2. If characters <= 8 bits Then
Append bit 1 to message OR Add 0x80

3. for each chunk
break chunk into sixteen 32-bit big-endian words w[i], 0 ≤ i ≤ 15
4. for i from 16 to 79
w[i] = (w[i-3] xor w[i-8] xor w[i-14] xor w[i-16]) leftrotate 1
a = h0
b = h1
c = h2
d = h3
e = h4
5. **for i from 0 to 79**
If 0 ≤ i ≤ 19 Then
f = (b and c) or ((not b) and d)
k = 0x5A827999
Else If 20 ≤ i ≤ 39 Then
f = b xor c xor d
k = 0x6ED9EBA1
Else If 40 ≤ i ≤ 59 Then
f = (b and c) or (b and d) or (c and d)
k = 0x8F1BBCDC
Else If 60 ≤ i ≤ 79 Then
f = b xor c xor d
k = 0xCA62C1D6
temp = (a leftrotate 5) + f + e + k + w[i]
e = d
d = c
c = b leftrotate 30
b = a
a = temp
h0 = h0 + a
h1 = h1 + b
h2 = h2 + c
h3 = h3 + d
h4 = h4 + e
hh = (h0 leftshift 128) or (h1 leftshift 96) or (h2 leftshift 64) or (h3 leftshift 32) or h4

4. HMAC Algorithm Pseudo code :

Input: key, message
Output: Hash Valuehh
Processing:

1. If (length (key) > blocksize) Then
   key = hash(key)
   End If
2. If (length (key) < blocksize) Then
   key = key ‖ [0x00 * (blocksize - length (key))]
   End If
3. o_key_pad = [0x5c * blocksize] ⊕ key
4. i_key_pad = [0x36 * blocksize] ⊕ key
5. hh = hash o_key_pad ‖ hash(i_key_pad ‖message)

4. SYSTEM IMPLEMENTATION

Aim of system implementation is text data de-duplication check. Java jre 7 environment on windows 7 platform is used for implementation of this system. User system, P-CSP and S-CSP are independent systems generated in this process. A desktop application created using swing control using net beans IDE is known as user system. Web based systems S-CSP and P-CSP communicate with desktop client using http client facility. Apache tomcat - 7 is used for web server setup and mysql 5.6 is used for storing records. Multipart web service facility format is used for complete file data / file blocks transfer among entitles. Also json format is used for short message communication.

5. RESULTS

5.1 Experimental setup

For testing an application single nodes windows -7 system is used. Jdk-1.7 Apache-7 and mysql-5.6 is configured on same system. War files of S-CSP and P-CSP are deployed on tomcat.

5.2 Results

We have divided our system in 2 major domains

1. User interaction and data communication: In this section GUI for user and data Communication between client application with P-SCP and S-CSP is tested.

2. System business logic: Initially we have tested our system business logic algorithm. It includes

We have implemented system for file level de-duplication check as well as block level de-duplication check. Text files are checked for block level de-duplication whereas image files are tested for file level de-duplication.

File block creation: File data is divided into number block. We have varied file block size with different values as 1kb, 2kb, and 4kb. As we increase the block size probably of de-duplication also increases.

For Image deduplication check we have created sha tag for a complete image file and file encryption is done using AES algorithm.

File Size(mb)	TagGenTime	KeyGen Time	Enc Time	Dec Time
1	47	46	827	359
2	73	93	890	374
3	125	109	936	453
4	156	109	998	484

Table.1 Time estimation Values

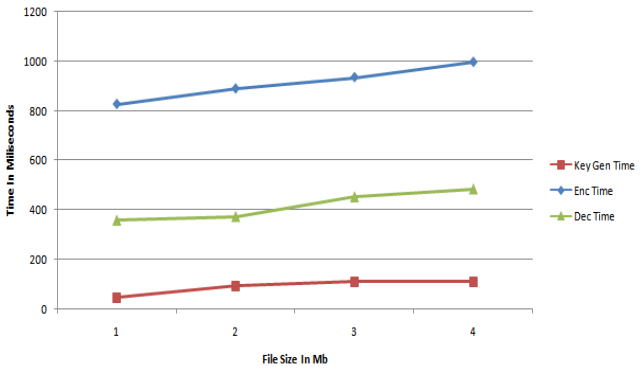

Fig.5 Time estimation Graph (a)

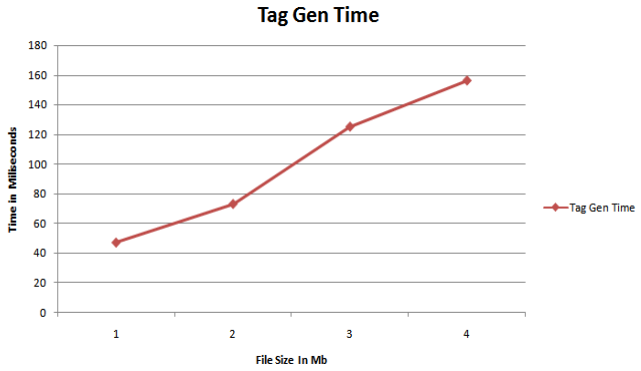

Fig.6 Time estimation Graph (b)

For text files, We have checked de-duplication for different block sizes like 2kb, 4kb, and 8kb. As we reduced the block size number of block increases and hence mapping time increases. Lower block size provide more accurate and précised de-duplication check results.

Following chart represents the block level file uploading time in detail.

File	tag gen time	file dedu time	convergent key time	tag gen time block	enc time for block	block upload time	block level dedu check
F1	35	166	1	164	164	10824	10824
F2	38	177	2	254	254	19812	19558
F3	45	200	2.5	380	380	30020	30400
F4	67	269	3	508	508	41148	44704

Table.2 Time estimation Values

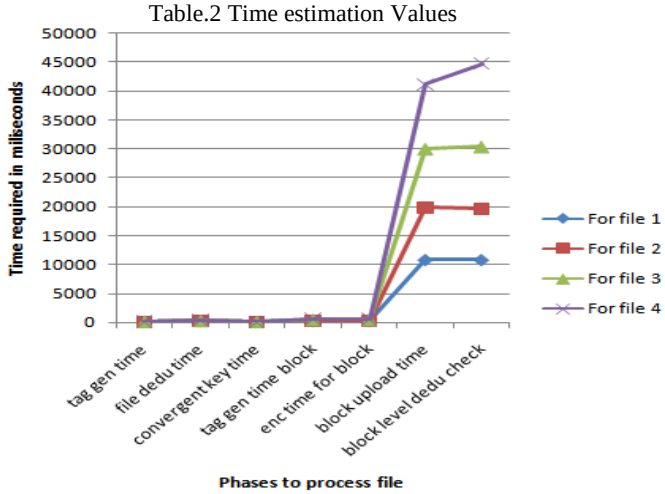

Fig.7 block level files uploading time

If file is already present then only proof of ownership program is executed and hence no upload time is required.

File	tag gen time	file dedu time	convergent key time	tag gen time block	enc time for block	block upload time	block level dedu check
F1	35	166	1	164	0	87	0
F2	38	177	2	254	0	88	0
F3	45	200	2.5	380	0	88	0
F4	67	269	3	508	0	89	0

Table.3 Time estimation Values

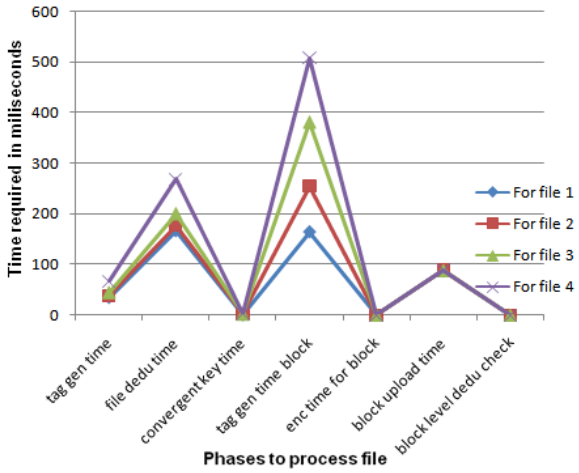

Fig. 8 File is Present

If file is partially present then only remaining blocks are uploaded to S-CSP and links are created for the file. We have tested this scenario for different cases where 25%, 50%, 75% file is already present on the server.
Following table shows the detailed description for partial file level de-duplication.

File	25%	50%	75%	100%
0.5mb	8118	5412	2706	87
1mb	14859	9906	4953	90
1.5mb	22515	15010	7505	95
2mb	30480	20574	10287	100

Table.4 Time estimation Values

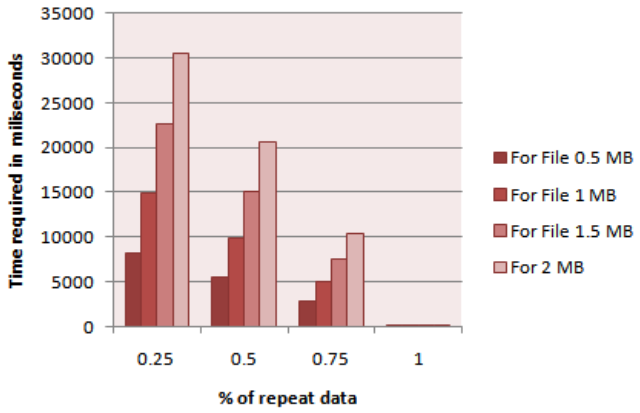

Fig.9 Duplication ratio

6. CONCLUSION

The project focuses on a technique to provide shared data security in cloud environment using data encryption. Effective usage of storage space at file level as well as block level is provided using de-duplication check. To support authorized duplicate check in hybrid cloud architecture new de-duplication constructions is provided, in which private cloud server is used for the duplicate check. This avoids multiple transaction of complete file also with the help of tag over network while checking de-duplication usage of bandwidth can be effective. The outsourcing of convergent key generation for encryption and key management is done at private cloud server. A relative addressing method in which relative file block address and its proper mapping logic maintained at two different sources P-CSP and S-CSP is introduced. Hacking and data predictions are blocked due to this process.

7. FUTURE SCOPE

Now a day's implementation of this technique on distributed system, using secret sharing instead of convergent key is being pursued. In future we work on de-duplication check for other than txt file.

8. ACKNOWLEDGEMENT

I would like to express my sentiments of gratitude to all who rendered their valuable guidance for this work. I would like to thank Dr. P. C. Kulkarni, Principal, GES's. R. H. Sapat College of Engineering Nashik, for providing me strong platform to develop my skills and capabilities.

I am sincerely thankful to Prof. C. R. Barde my guide and Prof. N. V. Alone, Head of Department, Computer Engineering Prof. A. S. Vaidya our PG co-ordinator for helping and guiding me with the topic and also providing me with adequate facilities, ways and means by which I was able to complete this paper.

9. REFERENCES

[1] Jin Li, Xiaofeng Chen, Mingqiang Li, Jingwei Li, Patrick P.C. Lee, and Wenjing Lou:"Secure De-duplication with Efficient and Reliable Convergent Key Management", *IEEE transactions on parallel and distributed systems*, vol. 25, no. 6, june 2014

[2] S. Quinlan and S. Dorward. Venti: "a new approach to archival storage". *In Proc. USENIX FAST*, Jan 2002

[3] P. Anderson and L. Zhang. "Fast and secure laptop backups with encrypted de-duplication". *In Proc. Of USENIX LISA*, 2010

[4] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer. "eclaiming space from duplicate files in a serverless distributed file system". *In ICDCS*, pages 617-624, 2002. S. Halevi, D. Harnik, B.

[5] Pinkas, and A. Shulman-Peleg. Proofs of ownership in remote storage systems. *In Y. Chen, G. Danezis, and V. Shmatikov, editors*, "ACM Conference on Computer and Communications Security", pages 491-500. ACM, 2011

[6] M. Bellare, S. Keelveedhi, and T. Ristenpart. "Dupless: Serveraided encryption for de-duplicated storage". *In USENIX Security Symposium*, 2013

[7] M. Bellare, C. Namprempre, and G. Neven. "Security proofs for identity-based identification and signature schemes". *J. Cryptology*, 22(1):1-61, 2009

[8] D. Ferraiolo and R. Kuhn. "Role-based access controls". *In 15th NIST-NCSC National Computer Security Conf.*, 1992.

[9] M. Bellare and A. Palacio. Gq and schnorr "identification schemes: Proofs of security against impersonation under active and concurrent attacks." *In CRYPTO*, pages 162-177, 2002

[10] M. Bellare, C. Namprempre, and G. Neven. "Security proofs for identity-based identification and signature schemes". *J. Cryptology*,22(1):1–61, 2009.

[11] S. Bugiel, S. Nurnberger, A. Sadeghi, and T. Schneider. "Twin clouds: An architecture for secure cloud computing". *In Workshopon Cryptography and Security in Clouds (WCSC 2011)*, 2011.

[12] K. Zhang, X. Zhou, Y. Chen, X.Wang, and Y. Ruan. Sedic: "privacyaware data intensive computing on hybrid clouds". *In Proceedings of the 18th ACM conference on Computer and communications security*, CCS'11, pages 515-526, New York, NY, USA, 2011. ACM.

[13] J. Xu, E.-C. Chang, and J. Zhou. "Weak leakage-resilient client-side de-duplication of encrypted data in cloud storage". *In ASIACCS*, pages 195-206, 2013

[14] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. "Proofs of ownership in remote Storage systems". *In Y. Chen, G. Danezis, and V. Shmatikov, editors, ACM Conference on Computer and Communications Security*, pages 491-500. ACM, 2011.

[15] W. K. Ng, Y. Wen, and H. Zhu. "Private data deduplication protocols in cloud storage",*In S. Ossowski and P. Lecca, editors, Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 441-446. ACM, 2012.

[16] R. D. Pietro and A. Sorniotti. "Boosting efficiency and security in proof of ownership for de-duplication", *In H. Y. Youm and Y. Won, editors, ACM Symposium on Information, Computer and Communications Security*, pages 81-82. ACM, 2012.