ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

Secure de-duplication in cloud using Convergent Encryption.

Komal Gaikwad, Anusha Rao, Mohana Upale, Neha Raut.

Department of Computer Science, Savitribai Phule Pune University, Pune.

Manuscript Info

Manuscript History:

Received: 14 December 2015
Final Accepted: 26 January 2016
Published Online: February 2016

Key words:
De-duplication,Convergent Encryption, Proof of Ownership (PoW), Multithreading.

Corresponding Author

Komal Gaikwad.

Abstract

Data de-duplication is a well-known technique to reduce storage space requirement and upload bandwidth in cloud storage by eliminating redundant data and keeping only one physical copy and referring other redundant data to that copy. In traditional encryption, identical data copies of different users leads to different cipher-texts, making de-duplication impossible. To overcome this limitation, convergent encryption is used. It encrypts or decrypts a data copy with a convergent key derived by computing cryptographic hash value of the data copy itself. Using De-key, de-duplication to the convergent keys is applied and secret sharing is leveraged

Introduction:-

In today's times the cloud technology is used by everyone as it has many useful features. The Google Drive, Dropbox, etc. are well known among the common users. The cloud allows the data to be accessed by user anywhere without having to carry it in a portable device. It also provides extra virtual space to save the data and allows the data to be available with many copies on different machines to which it has access to. Some organizations also use their private clouds to allow the employees to easily share, change and to store data more effectively as virtual space allows more storage space than physical space. According to the analysis report of IDC, the volume of data in the wild is expected to reach 40 trillion gigabytes in 2020 [1]. The technique of data De-duplication is used to avoid redundancy in storing data and make the data management scalable. Instead of keeping multiple data copies with the same content, de-duplication eliminates redundant data by keeping only one physical copy and referring other redundant data to that copy. The De-duplication may be done on different granularities such as, to either a whole file (i.e., file-level de-duplication), or a more fine-grained fixed-size or variable-size data block (i.e., block-level de-duplication).Today's commercial cloud storage services, such as Dropbox, Mozy, and Memopal, have been applying de-duplication to user data to save maintenance cost [2].

A user must trust third-party cloud providers to properly enforce confidentiality, integrity, and access control mechanisms against any insider and outsider attacks. However, de-duplication, while improving storage and bandwidth efficiency, is incompatible with traditional encryption. Specifically, traditional encryption requires different users to encrypt their data with their own keys. Thus, identical data copies of different users will lead to different cipher-texts, making de-duplication impossible.

The convergent encryption [3] provides a viable option to enforce data confidentiality while realizing de-duplication. In convergent encryption, data is encrypted with a convergent key, which is derived by computing the cryptographic hash value of the content of the data copy itself [3]. The users retain the key and the cipher-text is stored on the cloud. Since the encryption is deterministic, identical data copies will generate the same convergent key and the same cipher-text. This allows the cloud to perform de-duplication on the cipher-texts. The decryption can only be done by the users containing the convergent keys. The baseline approach explains how the convergent encryption can be applied.

Preliminaries:-

a. Traditional Encryption:-

To protect the confidentiality of outsourced data, various cryptographic solutions have been proposed in the literature [8]. Their idea builds on traditional (symmetric) encryption, in which each user encrypts data with an independent secret key. Some studies [6], propose to use threshold secret sharing [11] to maintain the robustness of key management. However, the above studies do not consider de-duplication. Using traditional encryption, different users will simply encrypt identical data copies with their own keys, but this will lead to different cipher-texts and hence make de-duplication impossible.

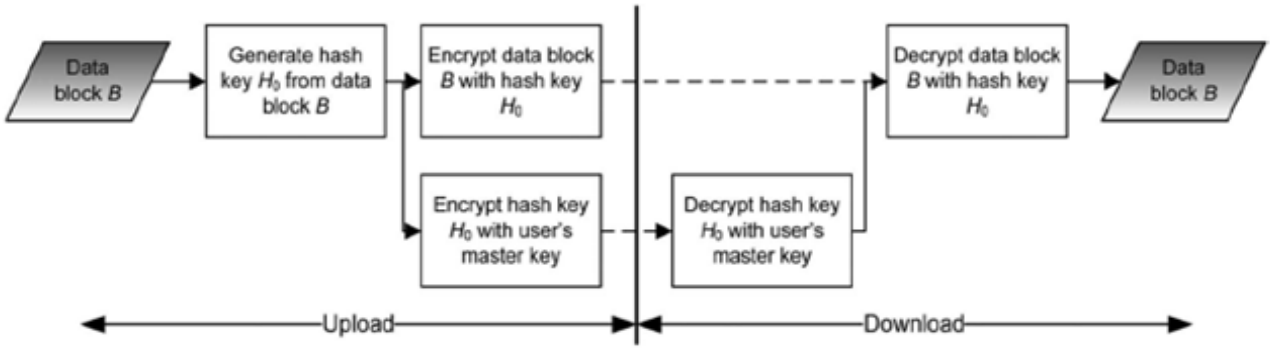

Figure1: Baseline Approach

1. Symmetric Encryption:-

A secret key, which can be a number, a word, or just a string of random letters, is applied to the text of a message to change the content in a particular way. This might be as simple as shifting each letter by a number of places in the alphabet. As long as both sender and recipient know the secret key, they can encrypt and decrypt all messages that use this key. There are mainly following main primitive functions:

- KeyGen(1^{λ}): The key generation algorithm generating key κ with the security parameter α.
- Encrypt(κ, *M*): *E* is the encryption algorithm that accepts the secret κ and massage *M* and then outputs the cipher text *E*.
- Decrypt(κ, *E*): *D* is the decryption algorithm that accepts the secret κ and encrypted format *E* to generate plain text *D*.

2. Convergent Encryption:-

If a conventional crypto-system was used to encrypt its files, then two identical files encrypted with different users' keys would have different encrypted representations. Then the system could neither recognize that the files are identical nor coalesce the encrypted files into the space of a single file, unless it had access to the users' private keys, which would be a significant security violation. A crypto-system called convergent encryption produces identical cipher text files from identical plaintext files, irrespective of their encryption keys. To encrypt a file using convergent encryption, a client first computes a cryptographically strong hash of the file content. The file is then encrypted using this hash value as a key. Along with this, *tag* is generated for each data copy which is used to detect duplicates. 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 [5].

b. Proof of ownership (PoW):-

A cryptographically secure and efficient scheme for a client to prove to the server, based on actual possession of the entire file instead of only partial information about it [10].This mechanism in PoW protects the security in client-side de-duplication. In this way, a client can prove to the server that it indeed has the file. De-key supports client-side de-duplication with PoW to enable users to prove their ownership of data copies to the storage server [7]. A cryptographically secure and efficient scheme, called a provable ownership of the file (POF) for a client proves to the server that it indeed has the file. An efficient goal is achieved by relying on dynamic spot checking, in which the

client only needs to access small but dynamic portions of the original file to generate the proof of possession of that file. This greatly reduces the burden of computation on the client and minimizes the I/O between the client and the server. At the same time, by utilizing dynamic coefficients and randomly chosen indices of the original files, this scheme mixes the randomly sampled portions of the original file with the dynamic coefficients to generate the unique proof in every challenge[7].

c. De-key:-

The concept of De-key is to efficiently and reliably maintain convergent keys. Its idea is to enable de-duplication and then distribute the convergent keys across multiple Key management Cloud service providers. Traditionally, the convergent keys are encrypted on per-user basis. But, De-key constructs secret shares on the original convergent keys and distributes the shares across multiple Key management Cloud service providers. The same convergent key is used by multiple users that share the corresponding block. This significantly reduces the storage overhead for convergent keys. [14][10]

d. Ramp secret sharing:-

De-key uses the Ramp secret sharing scheme (RSSS) [6] to store convergent keys. It is designed to protect single secrets based on a threshold number of users and can be termed as (K,n) threshold schemes. In these schemes a single secret χ, is split into n shares {S1, S2,…, Sn } and at least K shares are required for perfect reconstruction. However, K-1 or fewer shares do not have adequate information to reconstruct the secret either in the information theoretic sense or in the computational sense.

Related work:-

The data to be uploaded is checked with all the data present on the cloud using a comparator algorithm which will compare the Hash value generated from the text to the Hash values of the data stored on the cloud. The data is then encrypted by a convergent encryption algorithm resulting in the cipher text. Then the generated key is again encrypted with the user's master-key providing double encryption to the data. Then this data is then stored on the cloud[15].

But, the main issue with the Baseline approach(see Figure 1) [12]is that it is inefficient. Even if different users share the same data copies, they must have their own set of convergent keys so that no other user can access their files. All the convergent keys are encrypted with each user's master-key. Thus, the key management overhead increases as the number of users increases. Thus it can be said that the number of convergent keys being introduced linearly scales with the number of blocks being stored and the number of users. For Example, the Dropbox uses the SHA-256 to identify the duplicate copies of data. If a user is storing the 2TB of data in the blocks of 4KB then the total size of key storage for the particular user will be 16GB [4]. If there is another user that shares almost half of this data then this new user will have his own set of convergent Keys for each shared block. Thus, the increased number of users directly corresponds to the increased number of keys[10].

The other issue with the Baseline approach is that it is unreliable as it does not guarantee security. Each user must protect its own master-key. All the data recovery is dependent on the master-key, so if the master-key is lost accidentally the data stored cannot be recovered also the attackers can use the compromised data and leak it. In order to overcome these issues the concept of De-key is used. The *De-key* is used to manage the convergent keys efficiently and reliably allowing secure de-duplication.

Proposed model:-

Our main idea is to provide privacy as well as security to the user and allow efficient use of cloud space. The project is mainly associated with the information security, confidentiality and storage efficiency as well as processing efficiency. The key management, tag generation and other processes are implemented in parallel making the system efficient.

a. Avoids duplicate copies of data:-

Our system avoids existence of duplicate data to reduce storage space as well as upload bandwidth. This goal is achieved with convergent encryption and proof of ownership. It also manages the convergent keys derived with convergent encryption by avoiding duplicate keys for same data.

Tag Generation: System derives a tag for the data copy, such that the tag will be used to detect duplicate files. The tag will be generated such that, the tag for the files having same content uploaded by different users will be equal. For example, for input file F, the tag $T(F) = TagGen_{CE}(F)$ is computed and sent to the storage. Tag checking:

Whenever a file is being uploaded on the cloud the tag of that particular file is compared with the already existing tags by using a comparison algorithm.

b. Provides convenience to the user:-

If the user is uploading a file with duplicate content, the file will not be uploaded but the PoW of the user will be updated and the user will be informed that data is uploaded successfully. The user interface is very simple to understand and operate for the normal user thereby making the system convenient.

c. Provides security to data copies:-

Convergent Encryption: The convergent encryption process makes the data inaccessible to the hackers. Encrypting the data with convergent encryption reduces the significant security violation. Proof of ownership: It supports client-side de-duplication to enable users to prove their ownership of data copies to the storage server. PoW is implemented as an interactive algorithm run by a prover (i.e., user) and a verifier (i.e., storage server). The verifier derives a short value $\phi(M)$ from a data copy M. To prove the ownership of the data copy M, the prover needs to send ϕ' and run a proof algorithm with the verifier. The ownership is proved if and only if $\phi' = \phi(M)$ and the proof is correct.

d. Efficient processing of the data:

Tasks can be regarded as logical units of work, and threads are a mechanism that enables tasks (units of work) run asynchronously. Using multithreading for convergent encryption of the data that is to be uploaded on the cloud speeds up the processing and makes the system efficient.

System design:-

The System's first phase works on the client server architecture. The client sends the requests to the server to access the data stored on cloud. The client and server are connected by LAN connections. The server side performs the operation requested by the client and sends responses to the client over LAN network.

(see figure 1)At the back end of the server, cloud storage is used to store the data. The server performs operations like upload, open/download and delete on the cloud. The system provides information security by storing the data in encrypted form on the cloud. So each time when data is opened/downloaded by the user it is decrypted. The deletion block is used when the client wishes to delete the file uploaded. Thus, the system has 3 main blocks-

- Encryption Block
- Decryption Block
- Deletion Block

a. Encryption block:-

The encryption block creates cipher text that is stored in the cloud. It consists of Tag generation block, tag comparison block as well as encryptor with SHA and Master key. When the file is uploaded, the system checks whether a copy of the file exists on the cloud. It generates the tag(similar to Hash Value) and compares it with existing tags in the tag comparator. If the tag is unique then the file or data is then converted in cipher text using a function from SHA. Then the ownership of the file is given to the user when its master key is added to the PoW database.

b. Decryption block:-

Whenever the client seeks to open/ download the file, the decryption block gets activated. The decryptor block decrypts the data stored on cloud and converts it into plain text. The PoW checker and decryptor are situated in the decryption block. The PoW checker checks if the user requesting is the owner or ordinary user as only owner has the right to download the file. So the option of download will be accessible to the owner only. Others can only open the content according to owner specifications. The decryptor block then decrypts the cipher text to plain text.

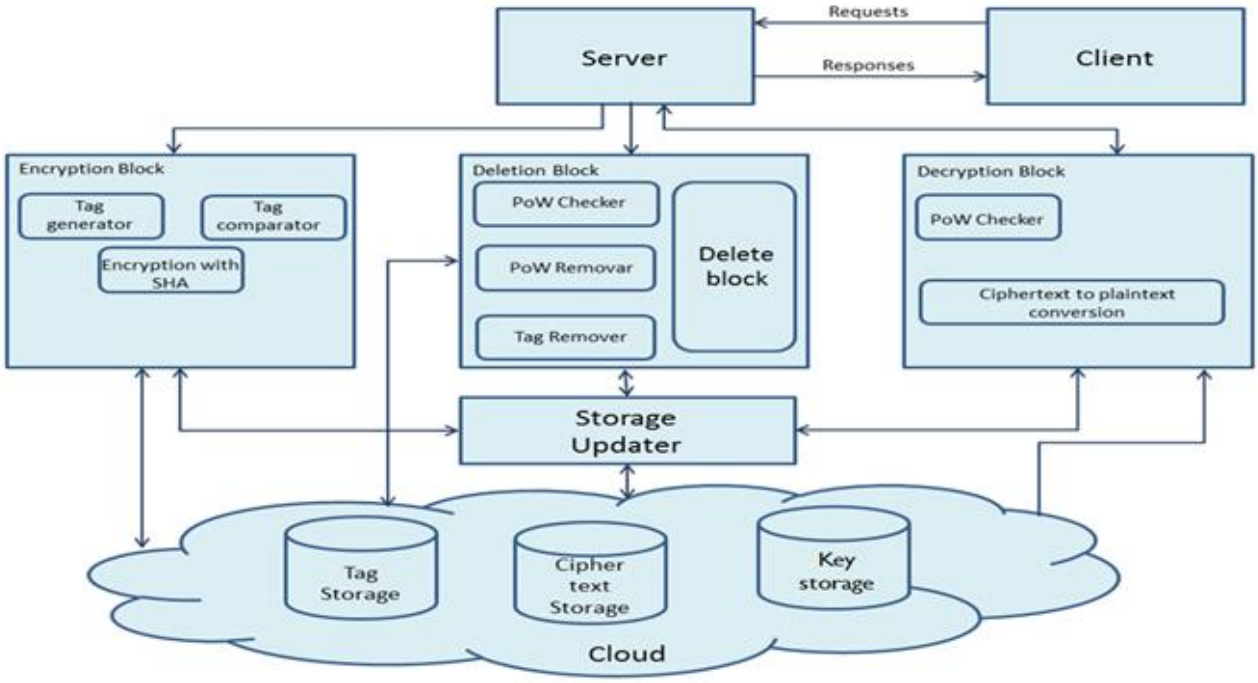

Figure 2: System Architecture

c. Deletion Block:-

The deletion block also consists of the PoW checker. As only the owners of the file will have the permission to delete the file. Same like in decryption block PoW checker Checks the ownership rights. Then if more than one user has PoW rights then only the rights of this user are removed and the file is kept as it is on the cloud. If only one user is the owner then the file is removed from the cloud storage along with the tag of the file and PoW permissions.

The cloud stores encrypted data as well as database for Tags and PoW permissions to allow easy access to them. The storage updater holds the information of the storage space remaining on the cloud and is updated every time when an upload or delete operation is performed on the cloud.

Algorithm:-

a. Uploading the File:-

Step 1: Generate the tag for the block B

T=TagGen(Block B)

Step2: While (Tag_storage!=NULL)// where tag_storage is the pointer to the tag Storage

{

If(T==Tag) then

Locate the corresponding data from cloud

Flag=1;

If(Flag==1) then

Break;

}

If(Flag==0)

{

$\kappa = KeyGen(1^{\lambda})$

E= Encrypt(κ, *B*)

//generates the encrypted form E with convergent key κ for data B.

Upload the generated tag value to the tag storage

}

If(Flag==1) then

Link that user to already existing data

Step 3: Print" Data is uploaded"

b. Downloading the File:

Step 1: Accept the file name from the user

Step 2: PoW checker checking the ownership

If (PoW is proven) then

D=Decrypt(κ, *E*)

//where D= decrypted form of the encrypted format E

Step 3: Send file D to the user

Merits:-

Data de-duplication is used to reduce the amount of storage space and save bandwidth. It also protects confidentiality of sensitive data while uploading and downloading the files on the cloud. Using the POW, we identify attacks that exploit client side de-duplication and attempt to identify reduplication. The combination of convergent encryption and multithreading provides efficiency.

Limitations:-

This De-duplication system will work with maximum efficiency for text files only. Also, enhanced Dynamic whole file De-duplication (DWFD) for space optimization in private cloud storage is not sufficiently used in development of chunk level de-duplication and block level de-duplication.

Conclusion:-

The output of the system would be a centralized authentication portal which would give access to cloud storage to the users registered. We make use of de-duplication to reduce storage space. The system is expected to provide access control policies and should not be easily penetrable.

Acknowledgements:-

In spite of our efforts taken in this project, it would not have been possible without the kind support and help in every aspect. We would like to extend our sincere thanks to all the professors who supported us. We are thankful to Prof. M.V Marathe for their guidance and constant supervision as well as for providing necessary information regarding the project. Our thanks and appreciations also go to our colleagues in developing the project and people who willingly helped us out with their abilities.

Result and Discussion:-

As discussed above De-duplication on cloud using Convergent Encryption is simple and efficient. This approach avoids the storage of duplicate copies at the initial stage itself. It also preserves the confidentiality of the user data.

References:-

1. **Akanksha V. Patil1, Navnath D. Kale2, (May 2013).**"A Secure Authorized Hybrid Cloud Distributed Key Generation for Encrypted Deduplication of Data" in International Journal of Science and Research (IJSR).
2. **D.Harnik, B. Pinkas, and A. Shulman-Peleg,(Nov./Dec. 2010).** ''Side Channels in Cloud Services:Deduplication in Cloud Storage,'' IEEE Security Privacy.

3. **Mohamad Thoufeeq, Dr. Sharmeela Sankar, Dr. M. Sandhya,** "Secure Authorized De-duplication Process in Hybrid Cloud ", in International Journal of Advance Research in Computer Science and Management Studies.
4. **M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. Weippl,(2011).** ''Dark Clouds on the Horizon:Using Cloud Storage as Attack Vector and Online Slack Space,'' in Proc. USENIX Security.
5. **John R. Douceur, AtulAdya, William J. Bolosky, Dan Simon, Marvin Theimer,** "Reclaiming Space from Duplicate Files in a Serverless Distributed File System", Microsoft Research.
6. **G.R. Blakley and C. Meadows,(1985).** ''Security of Ramp Schemes,'' in Proc. Adv. CRYPTO, vol. 196, Lecture Notes in Computer Science, G.R. Blakley and D. Chaum, Eds.
7. **Chao Yang, JianRen and Jianfeng Ma (School of CS, Xidian University),(2013).** "Provable Ownership of File in De-duplication Cloud Storage" in CISSS, available at {chaoyang, jfma}@mail.xidian.edu.cn.
8. **Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, (May 2011).**''Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing,'' IEEE Trans. Parallel Distrib. Syst.
9. **Jin Li, Yan Kit Li, Xiaofeng Chen, Patrick P.C. Lee, and Wenjing Lou, , (MAY 2015).** "A Hybrid Cloud Approach for Secure Authorized Deduplication" in IEEE Trans. Parallel Distrib. Syst.
10. **Jin Li, Xiaofeng Chen, Mingqiang Li, Jingwei Li, Patric P.C. Lee, and Wenjing Lou, (JUNE 2014).** "Secure Deduplication with Efficient and Reliable Convergent Key Management", in IEEE Trans. Parallel Distrib. Syst.
11. **Ronald Cramer, Serge Fehr**, "Reducing the Share Size in Robust Secret Sharing ", Mathematisch Instituut of Universiteit Leiden.
12. **N.O.AGRAWAL, Prof Mr. S.S.KULKARNI , (November 2014).** "Secure Deduplication And Data Securit With Efficient And Reliable CEKM" in IJAIEM.
13. **BogaVenkatesh, Anamika Sharma, Gaurav Desai,DadaramJadhav, (November 2014).** "Secure Authorised Deduplication by Using Hybrid CloudApproach" in International Journal of Innovative Research in Advanced Engineering (IJIRAE).
14. **Mrs. D.S.ArulMozhiArasi, Vaishali R, Saranya M, March(2015)**, " Secret sharing of Convergent Keys to Third Party Concept of Dekey" in International Journal of Research in Computer and Communication Technology.
15. **G. Prashanthi, Z.Shobarani, April(2015).** "A hybrid Cloud Approach for Secure Authorised De-duplication", in International Journal of Innovative Research in Computer and Communication Engineering.