International Journal of Future Generation Communication and Networking
Vol. 13, No. 1, (2020), pp.824-834

An Optimized Data Duplication Strategy for Cloud Computing: Dedup with ABE and Bloom Filters

Nipun Chhabra[a*], Manju Bala[b]

[a]*Ph.D. Research Scholar, I.K.Gujral. Punjab Technical University, Kapurthala (Punjab), India.*
[b]*Director, Khalsa College of Engineering & Technology, Amritsar , (Punjab), India.*

Abstract

In the present era, cloud computing has become very popular mainly because of its ease of access, unlimited data storage and on demand payment facility. Cloud's elastic provisioning capability gives access to additional storage space whenever required therefore many organizations prefer to store their data in cloud as it is readily available anywhere. It has been observed most of the time multiple copies of the same files/data are uploaded by the user leading to the loss of expensive storage and reduced bandwidth. To address this issue many data deduplication techniques were rendered which remove redundant information from any dataset using compression techniques. Deduplication releases a lot of free storage space when it is performed over big data volumes reducing the amount of data to transmit across the network and can save significant money in terms of storage costs, bandwidth cost and backup speed. Data deduplication techniques ensure that only one unique instance of data is retained on storage and any other user who tries to update the same data will be given a reference or pointer to the originally stored copy. In this paper a novel technique has been proposed for secure deduplication using probabilistic data structure which is memory and time efficient, known as bloom filters. Work pertaining to the use of Attributes based encryption along with Bloom Filters for Deduplication, is not available. Results indicate that the proposed algorithm has excelled existing traditional encryption with hashing technique for deduplication and it has been empirically proven with experimental/simulated results in this paper.

Keywords*: Deduplication, Access Control on encrypted data, Probabilistic Data Structure, Attributes based encryption along with Bloom Filters, Hash Function, Bloom Filters, Cloud Computing, Cloud Data Storage, Optimized Data Duplication, Storage Optimization in cloud computing.*

1. Introduction

Data have become a key factor for scientific research, public administration and an ever-growing digital economy. Development of promising technology such as Big Data greatly depends on large quantity of data being fed from all domains and users across the world. Storage and security are two major concerns of modern day organizations who store data in cloud data server. With the increased exchange of data from cloud data servers, security and privacy of data is compromised at many stages. For security purpose data is encrypted using advanced encryption techniques before uploading on cloud servers which keeps the data safe from unauthorized access [1][2]. It has been observed that many of the times users/organizations are placing multiple copies of their data occupying expensive storage on cloud data servers which not only reduces the available space but also consumes network bandwidth unnecessarily. Researchers have shown that removing this redundancy can optimize the available storage capacity to a large extent, hence increases the productivity of cloud computing platform. As a solution to this issue various data compression techniques were rendered to remove the redundant data known as Data Deduplication [3]. Every time a new file is stored in cloud

ISSN: 2233-7857 IJFGCN

storage a unique hash code is generated and stored in a hash table, if the file already exists at cloud server it gets detected through its hash code and the file is not stored again in cloud instead a reference pointer is given. In Hashing, variable length messages are accepted as Input and fixed length unique strings are produced as hash codes and are stored in hash table. No two files can have the same hash code. Depending upon the hashing algorithms, where 32 or 64 bytes hash code is generated and in MD5 16 byte hash code is generated, bloom filters make use of 3 or 4-byte which is far smaller than the hash codes. Therefore it was decided to delve out more possibilities in deduplication using bloom filters. In this paper, a novel arrangement has been proposed using bloom filters for deduplication coupled with Attribute based encryption for enhanced access control.

2. Literature Survey

Literature survey has been categorized on following two major aspects in our research work.

Security: Researches show that encryption of data is mandatory to secure sensitive/valuable data on the cloud environment. Encryption is the technique of encoding the plain data using some key to a cipher text which can be decoded using some specific key to an understandable plain text. The choice of key depends upon the type of encryption and decryption technique opted for encoding the data. In symmetric encryption the same key and in asymmetric encryption, different key is used. To make the data transaction even more secure it is suggested to implement identity based or Attribute based access control mechanism on encrypted data. (K-ABE and CP-ABE)

Storage**:** Storage is one of the most popular utility offered by cloud computing. Organizations/users are creating/uploading multiple copies of important data leading to the wastage of precious storage space. To overcome this problem deduplication techniques were employed. Deduplication is commonly known as data compression techniques, which removes the redundant data. Deduplication techniques make use of hashing to insert or look-up for any data file. This deduplication technique is further improved by implementing it with time and space efficient bloom filters.

TABLE 1. INFERENCE TABLE

Sr. No.	Author	Technique	Findings
1	Shamir et al.[4]	Unique cryptographic scheme	Encryption process uses unique identity of recipient
2	Sahai A et al.[5]	Attribute-based Encryption (ABE) scheme	This technique provides a promising strategy for encryption and access based restriction on data.
3	Goyal et al.[6]	Fine-grained access control ABE technique	Additional security to the access of data.
4	Bethencourt et al.[7]	System based on CP-ABE	Private Key based on user's attributes and third party which is encrypting data specifies the policies for decryption.
5	Lewko et al.[8]	Decentralized multi authority CPABE	User's attributes can be found easily through global identifier.

ISSN: 2233-7857 IJFGCN

6	Ostrovsky et al.[9]	Access structure with key policies	It was based on non-monotone formulas, having additional computational overhead.
7	Chen et al.[10]	Secure convergent encryption key	Ensures data privacy.
8	Bellare,M et al.[11]	Message Locked Encryption(MLE)	The key for encryption and decryption is derived from message.
9	Bellare et al.[12]	DupLESS Server aided encryption.	Modified convergent encryption is sed. Server side deduplication.
10	Hur, J. et al[13]	Server-side deduplication scheme, dynamic ownership management with enhanced security	Data integrity against any tag inconsistency attack
11	Liu, J et al[14]	PAKE-based protocol two parties can privately compare their secrets and share the encryption key	First single-server scheme that enables cross-user deduplication of client-side encrypted data
12	Noha MM et al[15]	Combining hash function with encryption Algorithm for security in storage	To enhance Security in the cloud data storage; a hash function with encryption algorithm can be used
13	Li, J. et al[16]	Dekey , implemented with Ramp Secret Sharing Scheme	Users need not manage any keys on their own but instead securely distribute the convergent key shares across multiple servers.
14	Li, J.et al[17]	Authorized data deduplication	Protect the data security by including differential privileges of users
16	Jan Stanek. et al[18]	Differentiates data on the basis of popularity.	Semantic security for unpopular data and provides weaker security and better storage and bandwidth benefits for popular data
17	Jing Chi. et al[19]	Bloom filter is space-efficient and false prediction	It saves space and easy to implement, optimal number of hash function minimizes the false prediction
18	Yi, Lu. et al[20]	Variable-length signatures enables Bloom filters to perform flow deletions	Alternative to expensive CAM or hash table lookups
19	B. TIRAPATHI REDDY et al[21]	Hash functions and probabilistic data structure to ensure the ownership	Work proceeds in reducing the false positivity and provide efficient cloud storage system.
20	Zhang. et al[22]	Bloom filter array (BFA) for data deduplication in	Effectively alleviate the pressure of storage and network transmission, raise

ISSN: 2233-7857 IJFGCN

		backup system	the rate of data to be deleted and ensure higher data deduplication speed.

From the above Table 1, it is evident that though there are many deduplication techniques available in prevalent times still they lack in the performance in the one way or the other. Literature Survey shows that efficient deduplication techniques can optimize the available storage to store more data at the same cost. Further it has been observed that the available deduplication strategies can be optimized using superlative techniques to improve the lookup time while searching the data in dataset and privacy/security of the data can be enhanced using better access control mechanism. So, as the outcome of this, it was decided to consider ABE for encryption and Bloom Filter for checking membership in any dataset.

Rest of the Paper is organized as follows; Section-3 discusses the working of existing system and elaborates the proposed work. Section-4 apprises the techniques used in proposed system and methodology. Section-5 discusses the simulation results. Section-6 concludes the proposed work and suggests the future scope of the said research work.

3. Existing System

In the existing system the data is encrypted on the client side before uploading on cloud server and on the server side hashing technique is implemented for deduplication of data on cloud storage, as explained below.

A. Client side:

A user can be a new user or an existing one seeking access to the cloud data, registers itself on client side. After the login of user, if the user is authenticated by the server then the file which is to be uploaded is first encrypted then sent to the cloud data server.

B. Server side:

When some Encrypted file is received at the cloud data server, the hash code is calculated/generated for it. This hash code is searched and verified from the hash table in which the Hash codes of all existing files are stored. If the code matches with any of the existing Hash Table value then it implies that the file is already present in the cloud storage. So, the user who tried to update the file to the cloud server is granted with the index/reference of the existing file in the Hash Table that acts as a unique identifier of the file. Though, the data is encrypted on client side before uploading on the cloud, it has been observed that during data transactions, there are many chances that it can be overheard or accessed by any intruder who may possess the key which is a potential threat to data privacy and security.

Secondly, when the data is uploaded on the server side the hash code generated for the received file is 16/32 bytes long and it consumes more memory space and for examining the membership of any file in the dataset, it is time a consuming process which ultimately slows down the lookup operation [23].

To overcome the shortcomings of the existing system and to improve storage with optimized deduplication strategy a novel arrangement has been proposed which utilizes attributes based encryption for secure access control coupled with Bloom filters for improved lookup operation.

ISSN: 2233-7857 IJFGCN

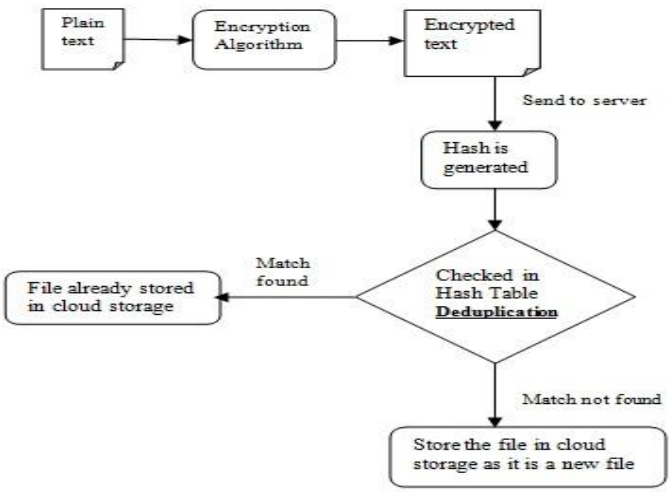


Fig No.1: Diagrammatic representation of the Existing System

4. Proposed System

To achieve the data security and optimum cloud storage an optimized deduplication technique in which Attribute Based Encryption (ABE) is integrated along with the bloom filters has been presented in this paper.

A. Attributes Based Encryption:

The encryption is done using a master key which is the combination of user key and user attributes and decryption takes place only when attributes of data matches the structure of access with the user's private key. It is a process that combines the access policy with keys for each user ***(Key, Policy)***.

B. Bloom Filter:

Bloom filter is a probabilistic data structure which is space efficient and time efficient. It is used to check the membership of an element in a dataset in relatively lesser time than hashing. During lookup operation, if an element is found in the dataset then corresponding bit position is marked with value 1 otherwise vale 0 is used for an element not present in the dataset. It has been observed, sometimes when element is not present in the dataset then also it returns one which is called false positive but false negatives are not possible. In bloom filters a group of hash functions are used to map elements into an array of bits. A bloom filter requires a bit array B[] initialized to zero. To store any data item using bloom filters, K hash functions are computed ranging from 1 to m and these vales refers to a single bit of m bit array B[m]. Consider two bit strings X and Y, which are to be programmed using bloom filter having hash functions k=3 and the size of bit array m=16. As shown in the fig no 2 (given below) overlapping bit patterns are also possible in bloom filters.

Procedure 1: To insert an element using bloom filter

```
Bfadd(x)                                    //bloom filter function to insert element 'x'
For i=1 to k,                               //iterate all hash functions in 'i' (h1(x), h2(x), h3(x))
        J=h_i(x);
        If B_j==0 then                      //it is empty insertion can be done
                Set B_j=1;
        End if
End For
```


ISSN: 2233-7857 IJFGCN

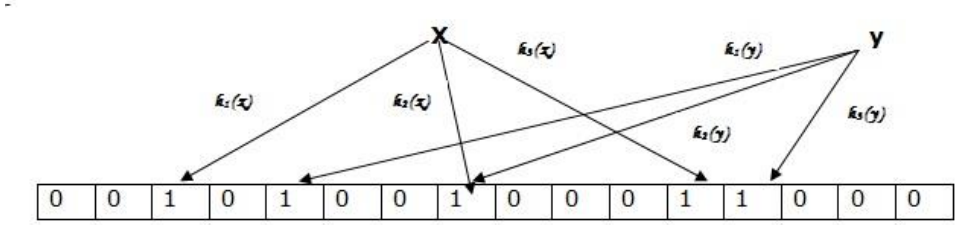


Fig No.2: Diagrammatic representation of inserting string X and Y using bloom filter, each having 3 hash functions

Procedure 2: To search an element using bloom filter

```
Bfsearch(x)                          //bloom filter function to search element 'x'
For i=1 to k, do                     //iterate all hash functions in 'i' (h1(x), h2(x), h3(x))
        J=hi(x);
        If Bj==0 then
                Element 'x' is present
        else
                element 'x' is not present
        End if
End For
```

In procedure 2 the implementation process for searching an element in a bloom filter is given. The bits are checked in 16-bit array at corresponding positions of hash functions. If all the corresponding locations are having bit 1 then the element is declared to be present in the dataset, but if at least one of the k bits is having 0 then it is considered element is not present.

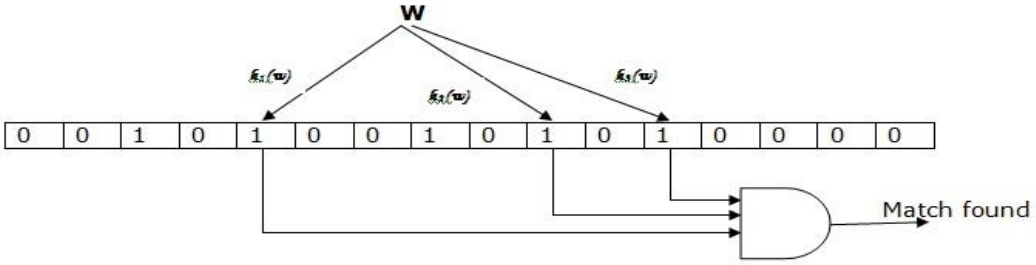
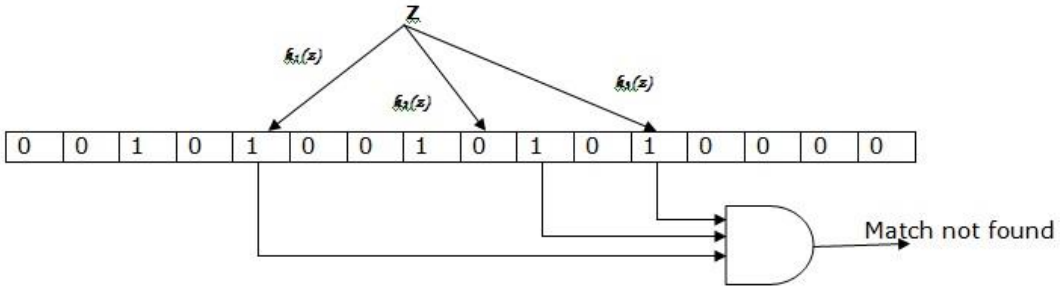


Fig No.3: Diagrammatic representation of searching W using bloom filter



ISSN: 2233-7857 IJFGCN

Fig No.4: Diagrammatic representation of searching Z using bloom filter

C. Client Side:

A user logs-in at client side with its credentials. After successful login, a key is generated using the user attributes. Then an authentication process is carried out. If the storage cloud authorizes the user then a file (f_1) which is to be uploaded on the storage is selected. After selection, it is encrypted at the client side using Attribute based encryption $E(f_1)$.

D. Server Side:

The hash key is generated for the encrypted file, if the generated hash key is possibly available in the bloom filter, bloom filter act as a filtration method i.e. that informs a data element may be in a set, or exactly it is not (false positives) It allows only the unique file to be uploaded into cloud storage or otherwise gives a possibility of the file to be already present in the cloud storage. The key advantage of using bloom filters is to save the computational time.

To test the presence of any element in a set of data, provide it to the hash functions to get *k* bit positions.

a. If any of the bits at these positions is 0, the element is not present in the dataset and the message of new file is displayed which can be uploaded in cloud storage.
b. If all the bit positions are 1, then the element may be present in the dataset and the message "file already stored" is displayed.

This way deduplication is checked and valuable storage is saved from storing the copies of existing files.

Algorithm for Optimized Deduplication using Bloom Filters with ABE

Phase I: Encryption and Deduplication
Step 1: User 'A' logs in using his credentials. // For master key generation these attributes are used
Step 2: A key 'K' is generated. //for Authentication of user.
Step 3: A file 'f' is selected for encrypted uploading on to cloud server.
Step 4: The file f is encrypted E(f) using Master Key(MK) // combination of K and user Attributes
Step 5: This encrypted file E(f) is transferred to cloud server and then a hash value is computed, let say H(E(f))
For j=1 to k (number of hash functions)
If $H_j(E(f))==0$ then
File not present
Else if $H_j(E(f))==1$ then
File already present in cloud storage
Step 7: Exit.

Phase II: Accessing data
Step1: Customer B wants to access any file f in Cloud storage
Step2: File f is selected and a request for access is sent to user A through Cloud server.
Step3: If user A grants access to B on file f, then a random key RK is generated and sent to Customer C. // which checks the authentication of customer
Step4: Reference of the key RK and Customer id is stored against the requested file f.
Step 5: Exit.

ISSN: 2233-7857 IJFGCN

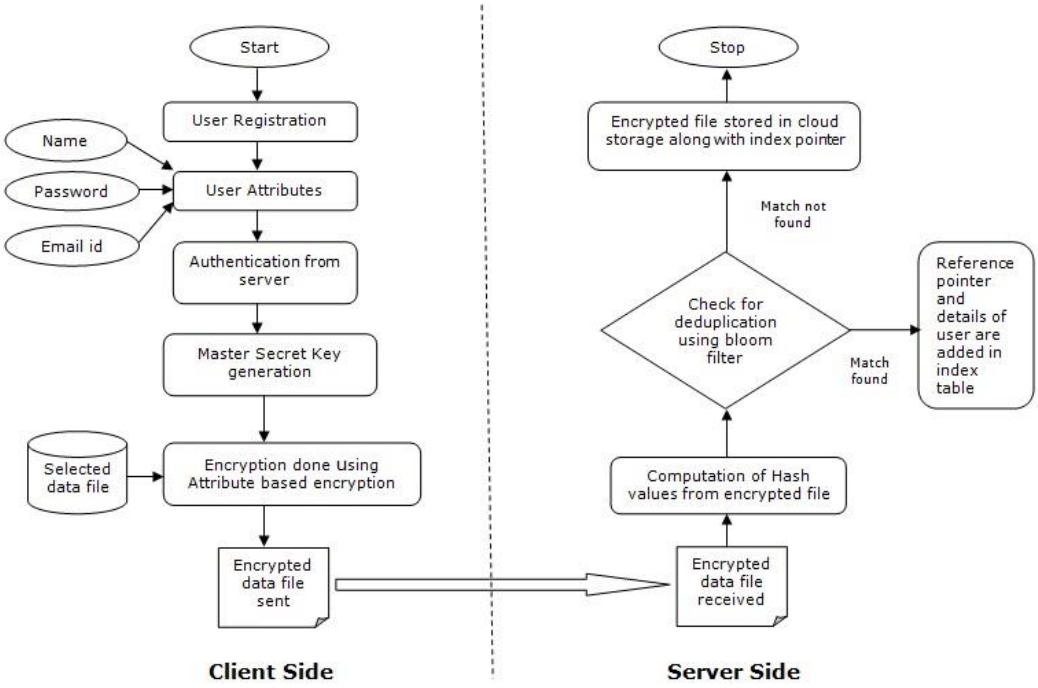


Fig No.5: Diagrammatic representation of the Proposed System

E. Methodology:

1. *Registration Phase:* In this phase a user interested in cloud access registers with its credentials and a user id is generated based for new or existing user
2. *Key Generation Phase:* A Key (KG) is generated using the unique attributes of the user.
3. *Authentication Phase:* in this phase, before uploading the file on the cloud storage, authentication process of the user is carried out by cloud server. If the user is authentic then only user can upload the file.
4. *Upload File:* A file (f) is selected which is to be uploaded in the cloud storage and the selected file is encrypted using Master Key (MK) which is the combination of KG and Attributes. This encrypted file E(f) is uploaded on the cloud storage.
5. *Hash Generation:* Encrypted file (E(f)) is received on the server side and a hash is generated H(E(f))
6. *Deduplication:* To test the presence of any element in a set of data, provide it to the hash functions to get *k* bit positions.
    a) If any of the bits at these positions is 0, the element is not present in the dataset.
    b) If all the bit positions are 1, then the element may be present in the dataset.

    If the generated hash key is possibly available in the bloom filter, there is a possibility of the file to be already present in the Cloud storage and file is not stored in cloud rather a reference pointer is given to the user.
7. *Storage in cloud:* If the Hash value is not present in the bloom filter then it is a new data and the File is uploaded in cloud and its reference is added into Hash function table.

5. Results and Discussions

The proposed system has been implemented in NetBeans using cloudsim 3.0.3 simulator having framework of java. The performance of the existing system can be compared with the proposed system on the basis of following metrics..

ISSN: 2233-7857 IJFGCN

a) *Memory space requirements.* As Compared to the hashing techniques, bloom filters use lesser memory space. The hash code produced in SHA-1 or SHA2 algorithm is 32 or 64 bytes, whereas bloom filters uses 3 or 4 byte filters which is very less as compared to secure hashes.

b) *Computational time for deduplication:* Bloom filter is a sequence of bit array where the presence of any element is indicated with 1 and absence is indicated by 0 bit. So, the time taken by bloom filter is very less as compared to hashing as in hash code lengthy comparisons are done for lookup operation.

The graph has been plotted using experimental results obtained by the simulation of the proposed and existing system shown below in Fig. No 6.

Fig. No 6. Line Chart Graphical Representation of the Experimental Results for the Lookup operation in Existing System vs. Proposed System

6. Conclusion

To achieve the data security and optimum cloud storage an optimized deduplication technique in which Attribute Based Encryption (ABE) is integrated along with the bloom filters has been presented in this paper. This arrangement had shown very good results as compared to the existing system in the lookup time of data in a dataset and has provided secure access to the user by utilizing user's credentials/attributes as a key for encryption of the data. The proposed system has been implemented in private cloud environment and in future more research can be done to implement this strategy for federated cloud environment.

References

[1] Wang, W., Li, Z., Owens, R., & Bhargava, B. (2009). Secure and efficient access to outsourced data. Proceedings of the 2009 ACM Workshop on Cloud Computing Security – CCSW '09. doi:10.1145/1655008.1655016

[2] Wang, Q., Wang, C., Ren, K., Lou, W., & Li, J. (2011). Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. IEEE Transactions on Parallel and Distributed Systems, 22(5), 847–859. doi:10.1109/tpds.2010.183 .

ISSN: 2233-7857 IJFGCN

[3] Chhabra, N., Bala, M. (2018). A Comparative Study of Data Deduplication Strategies. 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC). doi:10.1109/icsccc.2018.8703363

[4] Shamir, A. (n.d.). Identity-Based Cryptosystems and Signature Schemes. Lecture Notes in Computer Science, 47–53. doi:10.1007/3- 540-39568-7_5

[5] Sahai A., Waters B. (2005) Fuzzy Identity-Based Encryption. In: Cramer R. (eds) Advances in Cryptology – EUROCRYPT 2005. EUROCRYPT 2005. Lecture Notes in Computer Science, vol 3494. Springer, Berlin, Heidelberg.

[6] Goyal, V., Pandey, O., Sahai, A., & Waters, B. (2006). Attribute-based encryption for fine-grained access control of encrypted data. Proceedings of the 13th ACM Conference on Computer and Communications Security - CCS '06. doi:10.1145/1180405.1180418

[7] Bethencourt, J., Sahai, A., & Waters, B. (2007). Ciphertext-Policy Attribute-Based Encryption.2007 IEEE Symposium on Security and Privacy (SP '07). doi:10.1109/sp.2007.11

[8] Lewko, A., Okamoto, T., Sahai, A., Takashima, K., & Waters, B. (2010). Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. Lecture Notes in Computer Science, 62–91. doi:10.1007/978-3-642-13190-5_4

[9] Ostrovsky, R., Sahai, A., & Waters, B. (2007). Attribute-based encryption with non-monotonic access structures. Proceedings of the 14th ACM Conference on Computer and Communications Security - CCS '07. doi:10.1145/1315245.1315270

[10] Yan, Z., Wang, M., Li, Y., & Vasilakos, A. V. (2016). Encrypted Data Management with Deduplication in Cloud Computing. IEEE Cloud Computing, 3(2), 28–35. doi:10.1109/mcc.2016.29.

[11] Bellare, M., Keelveedhi, S., & Ristenpart, T. (2013). Message-Locked Encryption and Secure Deduplication. Lecture Notes in Computer Science, 296–312. doi:10.1007/978-3-642-38348-9_18

[12] Bellare, M., Keelveedhi, S., & Ristenpart, T. (2013). DupLESS: server-aided encryption for deduplicated storage.

[13] Hur, J., Koo, D., Shin, Y., & Kang, K. (2016). Secure Data Deduplication with Dynamic Ownership Management in Cloud Storage. IEEE Transactions on Knowledge and Data Engineering, 28(11), 3113–3125. doi:10.1109/tkde.2016.2580139

[14] Liu, J., Asokan, N., & Pinkas, B. (2015). *Secure Deduplication of Encrypted Data without Additional Independent Servers. Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15.* doi:10.1145/2810103.2813623

[15] Noha MM. AbdElnapi, Fatma A. Omara, Nahla F.Omran (2016). A Hybrid Hashing Security Algorithm for Data Storage on Cloud Computing

[16] Li, J., Chen, X., Li, M., Li, J., Lee, P. P. C., & Lou, W. (2014). Secure Deduplication with Efficient and Reliable Convergent Key Management. IEEE Transactions on Parallel and Distributed Systems, 25(6), 1615–1625. doi:10.1109/tpds.2013.284

[17] Li, J., Li, Y. K., Chen, X., Lee, P. P. C., & Lou, W. (2015). A Hybrid Cloud Approach for Secure Authorized Deduplication. IEEE Transactions on Parallel and Distributed Systems, 26(5), 1206–1216. doi:10.1109/tpds.2014.2318320

[18] Stanek, J., Sorniotti, A., Androulaki, E., & Kencl, L. (2014). A Secure Data Deduplication Scheme for Cloud Storage. Lecture Notes in Computer Science, 99–118. doi:10.1007/978-3-662-45472-5_8

ISSN: 2233-7857 IJFGCN

[19] Chi, J. (2009). Research and Application on Bloom Filter. Applied Computing, Computer Science, and Advanced Communication, 30–35. doi:10.1007/978-3-642-02342-2_5

[20] Yi Lu, Balaji Prabhakar, Flavio Bonomi. Bloom Filters: Design Innovations and Novel Applications

[21] B. Tirapathi Reddy, M. V. P. Chandra Sekhara Rao(2018). Filter Based Data Deduplication in Cloud Storage using Dynamic Perfect Hash Functions

[22] Zhang, J., Zhang, S., Lu, Y., Zhang, X., & Wu, S. (2013). *Hierarchical Data Deduplication Technology Based on Bloom Filter Array. Proceedings of the International Conference on Information Engineering and Applications (IEA) 2012*, *725–732*. doi:10.1007/978-1-4471-4856-2_88

[23] Aditya, T., Baruah, P. K., & Mukkamala, R. (2011). Space-Efficient Bloom Filters for Enforcing Integrity of Outsourced Data in Cloud Environments. 2011 IEEE 4th International Conference on Cloud Computing. doi:10.1109/cloud.2011.40

ISSN: 2233-7857 IJFGCN