JCSE International Journal of Computer Sciences and Engineering Open Access

Survey Paper Vol.-7, Issue-3, March 2019 E-ISSN: 2347-2693

Secure Retrieval and Revocable Attribute-Based Encryption Scheme in Cloud Storage

M. Muthuselvi[1*], Pemi. P, Rajasree. S[2], Sowmiya. C[3]

[1,2,3]Dept. of Computer Science and Engineering, University College of Engineering, Nagercoil, TamilNadu, India

Corresponding author: pemi12898@gmail.com

DOI: https://doi.org/10.26438/ijcse/v7i3.675679 | Available online at: www.ijcseonline.org

Accepted: 11/Mar/2019, Published: 31/Mar/2019

***Abstract*--**Cloud security is the protection of data stored online from theft, leakage and deletion. Hierarchical attribute-based encryption scheme is first designed for a document collection. A set of documents can be encrypted together if they share an integrated access structure. Compared with the CP-ABE schemes, both the ciphertext storage space and time costs of encryption/decryption are saved. Then, an Index Structure named attribute-based retrieval features (ARF) tree is constructed for the document collection based on the TF-IDF model and the documents attributes. A depth-first search algorithm (DFS) for the ARF tree is designed. It is difficult to search the large collection of documents. To overcome the difficulties an IDDFS method is introduced.

Keywords-Cloud computing, document Retrieval, file hierarchy, attribute-based encryption.

I. INTRODUCTION

Cloud computing will collect and reorganize a large quantity of IT resources and apparently, the cloud servers will provide safer, flexible, various, economic and personalised services compared with the native servers. Cloud computing could be a large-scale distributed computing paradigm driven by reconfigurable computing resources will be rapidly provisioned and discharged with minimal management effort within the information centres. Increasing the outsourcing information user incessantly best owed sensitive information like government records, personal health records and photos etc., So information privacy and information loss are going to be increase. Once users source their private onto cloud, the cloud service provider able to monitor the communication between the users and cloud at can trust or untrusted. The cloud server leaks the information to unauthorized users or perhaps be hacked. To assure the secrecy, users sometimes encrypting the info before outsourcing it onto cloud; it brings the adult challenges to effective information utilization. Information homeowners additionally share their information to source cloud with a number of users, UN agency may wish to retrieve the files in an exceedingly given throughout session. Keyword based mostly retrieval is associate degree most popular technique for looking out the plaintext scenario, that users to retrieve relevant files supported keywords, however it's terribly difficult to retrieve the files in ciphertext. Improve the potency and practicability of searching introduces the relevant result files. Introduce the multi keyword top-k retrieval technique over encrypted cloud data, therefore a way to create the cloud do most work throughout the procedure of retrieval without escape of knowledge. A large number of searchable document encoding schemes are projected within the literatures, together with single keyword Boolean search schemes, single keyword ranked search schemes and multi-keyword Boolean search schemes. However, all these schemes cannot support effective, flexibility and efficient document search because of their straightforward functionalities. To solve those issues, Hierarchical attribute-based coding (H-ABE) schemes are applied to cloud storage services. For this purpose, there have been several of the schemes, proposed for coding. One of the scheme discuss about the Hierarchical Attribute-Based Encryption (ABE) more up the security. To our data, most existing schemes cannot support time economical retrieval over the documents that are organized underneath attribute-based access control mechanism. To support correct and economical document search over the encrypted documents, an advanced index structure is then made for the document assortment. Then to style of a ciphertext-policy attribute-based hierarchical document assortment encryption theme known as CP-ABHE. In this way, each the ciphertext Storage space and time prices of the encryption/ decryption are saved.

II. PROPOSED SYSTEM

Secure document storage and retrieval is important in cloud security. In the theme justify the way to firmly retrieve the documents from cloud is explained. The projected scheme will greatly decrease the storage and computing burdens. To map the documents of vectors in which each the keywords and associated attributes square measure thought of. The ARF tree is proposed to arrange the document vectors and support time-efficient document retrieval. Additionally, an Iterative Deepening depth-first search algorithmic program is meant. A thorough simulation is performed to illustrate the safety, potency and effectiveness of our theme. Specifically, the projected secret writing theme performs very well in each time and storage efficiency. In addition, the theme conjointly provides efficient and correct document retrieval method. In this theme knowledge owner encrypts the document along by ciphertext policy attributes based mostly secret writing and ARF tree is construct based mostly TF-IDF model. The IDDFS technique is employed to enhance the search potency. IDDFS is best fitted to a complete infinite tree. This may be phrased as every depth of the search co-recursively producing a stronger approximation of the answer, though the work done at every step is algorithmic. This is not possible with a standard depth-first search, that doesn't turn out intermediate results. In proposed scheme tend to arrange to style a fine-grained access management mechanism for the encrypted documents that additionally support economical document search. The search results of the questions are outlined because the top-k relevant encrypted documents with legal attributes.

III. ARCHITECTURE DIAGRAM

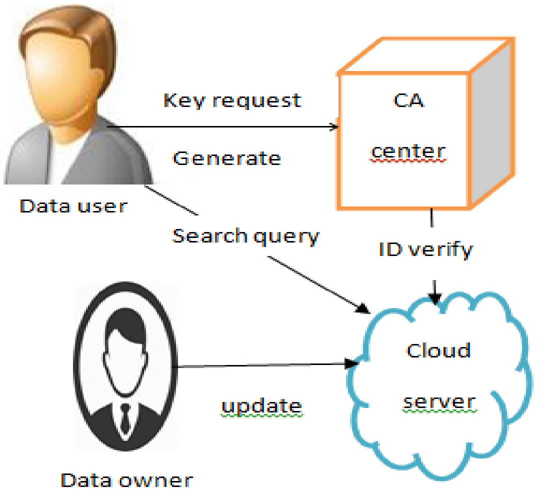

Fig1: System Model

Data owner collects the Documents and Encrypt the document assortment supported attributes. The attributes area units are encrypted by Hierarchical Attribute Based Encryption Scheme (H-ABE). Then, the encrypted documents area unit send to the cloud server and it store the encrypted document in cloud and it's to blame for capital punishment CA center document search supported ARF tree index structure.

Data User United Nations agency needs to go looking any document initially register to the Certificate Authority. If the user area unit authorized then the Certificate Authority generates the key to the User. The Data user send query to the Cloud Server and cloud server directly communicate to the Certificate Authority to check the legitimacy of the data user. Then, the cloud server searches the index structure to come back the search document to the data user. Once looking the corresponding encrypted documents area unit extracted from the encrypted document collection and sent to the info user. At last, the data user decrypts the documents by the secret key. With lawfulness checking, the data users United Nations agency didn’t register themselves in the CA center cannot search the interested documents through the cloud server. However, the safety of the system doesn’t greatly decrease while not this functionality and it is explained by the fact that the block information users cannot decrypt the documents came by the cloud server as a result of they don’t have the secret keys.

IV. INDEX STRUCTURE

The vector of a document consists of two elements as well as a normalized content vector associated an attribute vector. To build the content vector, every document is treated as a stream of keywords and use the normalized term frequency (TF) vector to quantize the documents. The TF-IDF model is employed to construct the index structure. To enhance the search potency of multi-keywords search method, a height balanced index tree named ARF tree is built supported the document vectors. TF*IDF is associate info retrieval technique that weighs a term’s frequency (TF) and its inverse document frequency (IDF). TF-IDF Model for Page ranking. TF-IDF stands for Term frequency-inverse document frequency.

The TF-IDF weight may be a weight typically used in info retrieval and text mining. Variations of the TF-IDF weighting scheme are typically employed by search engines in scoring and ranking a document's relevance given a question. This weight may be a statistical live wont to valuate however important a word is to a document in an exceedingly collection or corpus. The importance increases proportionately to the quantity of times a word seems within the document however is offset by the frequency of the word in the corpus. Variations of the TF-IDF weighting theme are typically employed by search engines as a central tool in marking and ranking a document's connection given a user query. One amongst the only ranking functions is computed by summing the TF- IDF for every question term; more sophisticated ranking functions are variants of this straightforward model. TF-IDF can be with success used for stop-words filtering in varied subject fields as well as text report and classification.

TF(t) = (Number of times term t seems in a document) / (Total range of terms in the document)

Inverse Document Frequency, which measures however vital a term is. While computing TF, all terms are thought about equally vital. But it's well-known that sure terms, like "is", "of", and "that", might seem heaps of times however have little importance. So we want to weigh down the frequent terms whereas rescale the rare ones, by computing the following:

IDF (t) = log_e(Total range of documents/ range of documents with term t in it).

Compute TF-IDF by multiplying an area component (term frequency) with a world component (inverse document frequency), and normalizing the ensuing documents to unit length. Formula for non-normalized weight of terms i in document j during a corpus of D documents.

$$weight_{i,j} = frequency_{i,j} * log_2 \frac{D}{document\_freq_i}$$

The TF-IDF worth will increase proportionately to the quantity of times a word seems in the document and is offset by the quantity of documents. TF-IDF is one among the foremost popular term-weighting schemes. Secure document storage and retrieval is one among the hottest analysis directions in cloud computing. Then, associate index structure named Attribute-based Retrieval Features (ARF) tree is built for the document collection supported the TF-IDF model and the documents attributes.

V. DOCUMENT COLLECTION

To propose a theme to assist enterprises to with efficiency share confidential knowledge on cloud server. The documents square measure collected based on the Hierarchical Attribute based mostly Encryption technique (H-ABE). Therefore cloud service supplier should give the trust and security, as there square measure massive amount of vital and sensitive knowledge stored on the clouds. For shielding the confidentiality of the hold on knowledge, the data must be encrypted before uploading to the cloud by victimization some cryptography algorithms. With the emergence of sharing confidential company knowledge on cloud servers, it's essential to adopt associate degree economical encryption system with a fine-grained access management to cipher outsourced knowledge.

Attribute-based cryptography (ABE) may be a public-key rule based mostly one too several encryptions that enables users to cipher and decode knowledge supported user attributes. HABE model consists of a Root Master (RM) that corresponds to the Third Trusted Party (TTP), Multiple Domain Masters (DMs) within which the top-ranking DMs correspond to multiple enterprise users, and various users that correspond to all personnel in associated degree enterprise. This scheme used the property of gradable generation of keys in HIBE theme to generate keys. This theme will satisfy the property of fine grained access management, scalability and full delegation. It will share data for users within the cloud in associate degree enterprises are environment.

VI. SEARCH FOR FILES

In Depth First Search is improved the search potency instead of the linear Search. However the DFS cannot support the infinite depth tree structure. To beat this drawback, the theme planned the Iterative Deepening Depth First Search (IDDFS). To avoid the infinite depth problem of DFS, decide to solely search till depth L, i.e. It have a tendency to don't expand beyond depth L. If resolution is deeper than L, Increase L iteratively.

Table.1

Criterion	BFS	DFS	IDDFS
Time	Yes	No	Yes
Space	O(b^{d+1})	O(b^{m})	O(b^{d})
complete	Yes	No	Yes

This inherits the memory advantage of Depth First search, and is best in terms of time complexness than Breadth First search. It additionally improved the search economical. Another challenge is looking the top-k relevant documents whose attributes area unit covered by the info users. It will more improve the search potency by operative the looking method in parallel. IDDFS is employed to examine if the goal is accessible from begin node. Therefore it come back type is Boolean. IDDFS is merely accustomed check, not come back the trail from begin node to goal. Therefore it had a tendency to don't maintain something like parent array (like in DFS).

IDDFS Algorithm:

```
// In IDDFS algorithm we are using DLS algorithm from earlier section IDDFS (root, goal)

{

depth=0

while(no solution)

  {
    Solution=DLS (root, goal, depth)
    Depth= depth+1
  }
return solution
}
DLS(node, goal, depth)
  {
If (depth>=0)
if (node==goal)
return node
for each child expand (node)
DLS (child, goal, depth+1)
}
```

IDDFS is meant to run DLS from zero → ∞, but we will write our IDDFS program to run DLS from zero → MAX_DEPTH. As a result of in real world it will never run something up to ∞.First code the DLS technique, then add the IDDFS technique that calls the DLS method. IDDFS is supposed to run in associate infinite house tree. However, the search completeness can't be warranted in DFS. In proposed theme, the similarity between a combine of documents is calculated based on each the content vectors and attribute vectors. The projected theme will always acquire the correct search results with search potency.

VII. PERFORMANCE EVALUATION

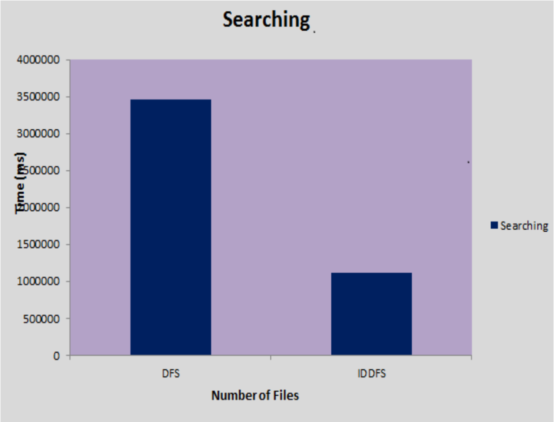

Fig. 2

In IDDFS, Depth First Search is applied along with the simplest depth limit. In each step, step by step it will increase the depth limit until the goal is found. It will increase depth limit from level zero, level 1, then level 2 until the shallowest goal is found at sure depth 'd'. repetitious Deepening Depth 1st Search are often enforced kind of like BFS because it explores an entire layer of new nodes at every iteration before going on to future layer. If it would like to avoid memory needs that are incurred in BFS then IDDFS are often enforced like uniform value search. The key purpose is to be use increasing depth limits rather than increasing depth limits, is that if it have such an implementation it's termed as repetitious length search. It guarantees completeness as the search doesn't stop till goal node is found.

CONCLUSION

The documents collection measure encrypted based on Hierarchical attribute based mostly encryption theme. To think about a brand new encrypted document retrieval state in which the data owner needs to regulate the documents in fine-grained level. To support this service, it have a tendency to first style a novel Hierarchical attribute-based document cryptography theme to encipher a set of documents along that share associate degree integrated access structure. Further, the ARF tree is planned to arrange the document vectors supported their similarities. At last, Iterative Deepening depth-first search rule is intended to improve the search potency for the data users that is extraordinarily vital for large document. In future, for attempt to the access structure of the document collection is generated in a greedy manner and it will check whether that can be further optimized to decrease the number of access trees.

REFERENCES

[1] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," IEEE Transactions on Parallel and Distributed Systems, vol. 25, pp. 222–233, Jan. 2014.

[2] C. Chen, X. Zhu, P. Shen, J. Hu, S. Guo, Z. Tari, and A. Zomaya, "An efficient privacy-preserving ranked keyword search method," IEEE Transactions on Parallel and Distributed Systems, vol. 27, pp. 951–963, Apr. 2016.

[3] H. Deng, Q. Wu, B. Qin, J. Domingo-Ferrer, L. Zhang, J. Liu, and W. Shi, "Ciphertext-policy hierarchical attribute-based encryption with short ciphertext," Information Sciences, vol. 275, pp. 370–384, Aug. 2014.

[4] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," IEEE Transactions on Parallel and Distributed Systems, vol. 27, pp. 2546–2559, Sep. 2016.

[5] Y. Guo, J. Li, Y. Zhang, and J. Shen, "Hierarchical attribute-based encryption with continuous auxiliary inputs leakage," Security and Communication Networks, vol. 9, no. 18, 2016.

[6] J. Li, X. Lin, Y. Zhang, and J. Han, "Ksf-oabe: Outsourced attribute-based encryption with keyword search function for cloud

storage," IEEE Transactions on Services Computing, vol. 10, no. 5, pp. 715–725, 2017.

[7] E. Luo, Q. Liu, and G. Wang, "Hierarchical multi-authority and attribute based encryption friend discovery scheme in mobile social networks," IEEE Communications Letters, vol. 20, pp. 1772–1775, Sep. 2016.

[8] Y. S. Rao, "A secure and efficient ciphertext-policy attribute-basedsigncryption for personal health records sharing in cloud computing,"Future Generation Computer Systems, vol. 67, pp. 133–151, Feb. 2017.

[9] S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, and W. Xie, "An efficient file hierarchy attribute-based encryption scheme in cloud computing," IEEE Transactions on Information Forensics and Security, vol. 11, no. 6, pp. 1265–1277, 2016.

[10] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," IEEE Transactions on Parallel and Distributed Systems, vol. 27, pp. 340–352, Jan. 2016.