International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 10 (2015) pp. 25967-25975
© Research India Publications
http://www.ripublication.com

Effective Storage and Search Optimization In Cloud

U.M.Ramya[1], G.L.K.Vasista[2] ,E.Jhansi[3], P. Sai Bharadwaj[4]
[1]*Assistant Professor, Department of Computer Science and Engineering, Amrita School of Engineering, Coimbatore*
[2,3,4]*Department of Computer Science and Engineering, Amrita School of Engineering, Coimbatore*
{ramya.u.m ,vasista.gutta, jhansi.enamala, saibharadwaj.pisipati}@gmail.com

Abstract

With the advent of Cloud computing as a dominant computing platform, data owners are provoked to handle complex and sensitive data by moving data from local sites to public cloud which provides great flexibility and low cost. For providing data discretion and privacy, unlike traditional data usage based on plaintext keyword search, sensitive data have to be encrypted before outsourcing. Thus, enabling searchable encryption service is of preeminent importance. Having the large number of data users and documents in cloud, it is necessary for the search service to provide optimal search with single keyword query and effective document retrieval based on relevance score of keywords. Related works on privacy preserving searchable encryption focus on single keyword search, Fuzzy keyword search or Boolean keyword search with or without similarity ranking. Cloud handles huge set of documents. When a cloud user posts a query to the cloud, it is known that the user would be retrieved with both relevant and irrelevant documents with which the user has to post-process the documents and sort out to get the relevant documents. This whole process is cumbersome to the user. In this paper, an architecture which provides an optimal search over encrypted cloud data has been proposed which allows the user to retrieve only relevant documents with respect to the query keyword. Elliptic Curve Cryptography (ECC) algorithm has been used for encryption of documents thereby securing the documents and Diffie-Hellman algorithm to check for authentication of key exchange. We have used inverted index to index each document. After the relevant documents are retrieved, we then rank each document based on TF-IDF scoring and provide the top K retrievals to the users.

Keywords: Diffie- Hellman, Inverted Index, tf- idf, Ranking, Cloud

Introduction

Memory storage is the primary usage of cloud computing. Cloud provides three services namely-1)Software as a service(SaaS) 2)Platform as a service(PaaS) 3)Infrastructure as a service(IaaS).The user uses these services based on their requirement. In this paper, we have used the Infrastructure as a Service(IaaS).

Cloud is an emerging technology for storing large amount of data. Cloud is a sensitive area where the documents can be easily accessed. In order to provide effective storage, the documents can be encrypted and stored in the cloud.

Encryption of the documents are done using Elliptical Curve Cryptography (ECC). The documents are encrypted and the encrypted documents are called as cipher texts. When the user queries, the documents are retrieved and it is decrypted by the user.

In order to provide an optimal search over cloud which allows the user to retrieve only relevant documents with respect to the query, we need to index all the documents before outsourcing it to the cloud to avoid retrieval of non-relevant documents [4]. Thus all the documents that have to be stored in the cloud have to be pre-processed (indexed).These indexes are stored in trusted local server. So whenever the user queries the cloud, it first goes to the local server and indexed data is referred [7]. We get the relevant document id's from the local trusted server which in turn are ranked and then sent to the cloud. The process of getting these relevant document id's will be explained in the later section. The cloud then sends the relevant documents to the user.

An untrusted cloud server can have multiple data users. The data users are allowed to access the documents only if they are authenticated [10].Authentication is the process for confirming the identity of the user.So,we have used a 3 way Diffie-Hellman algorithm to check for authentication. The user can query the cloud only if he is authenticated [2].

For a given collection of documents, indexing is defined as-Each word in the collection is called as a term, and corresponding to each term we maintain a list known as posting list, of all the documents in which the particular word appears [5]. Once we get the relevant document id's from the trusted local server we rank each document based on tf-idf scoring. Thus,the document with highest tf-idf score will be given higher priority. In later sections the working of each operation i.e, inverted index,encryption and decryption ,ranking,authentication is explained in detail.

Literature Survey

Indexing of the documents are done using Word level inverted index[4,5] in this paper as it provides information about position of each word within a document. There are many cryptographic algorithms available for authentication. Several papers have used RSA, DES, and Triple-DES which is not acceptably efficient. In this paper the authentication part is done using Diffie-Hellman algorithm because it is one way algorithm and the authentication channel provided is more secured[1].Retrieval of documents for the user is suggested by using Boolean search with the query which is not the effective way of

retrieving the documents[6].In this paper ranking of retrieved relevant documents is implemented using tf-idf, which is more efficient compared to other scoring algorithm such as vector space model[9], set oriented model[5]. In tf-idf,*rare keywords* are given more priorities than to frequent occurring words. Encryption and decryption of documents is done using ECC as it is computed in less time with fast processing rate. The ECC algorithm uses 256-bit key to provide the same security given by RSA of 3072-bit key. ECC is efficient with high complexity.

System Architecture

The three entities in our schema are: Admin, Local Trusted Server, and Untrusted Cloud Server. The admin is the one who stores the data in the cloud and has the authorization to go through the documents. Cloud server is the untrusted one where storage service is provided to the admin's data, where the admin can store their data in the form of cipher text. The documents of the admin are indexed and the indexed lists are stored in the local trusted server. The documents are stored in the untrusted cloud server. When the query is made by the user, the local server makes a document request to the untrusted cloud server, where on the ranking basis the retrieval of relevant documents is done and the retrieved documents are decrypted by the user [7].

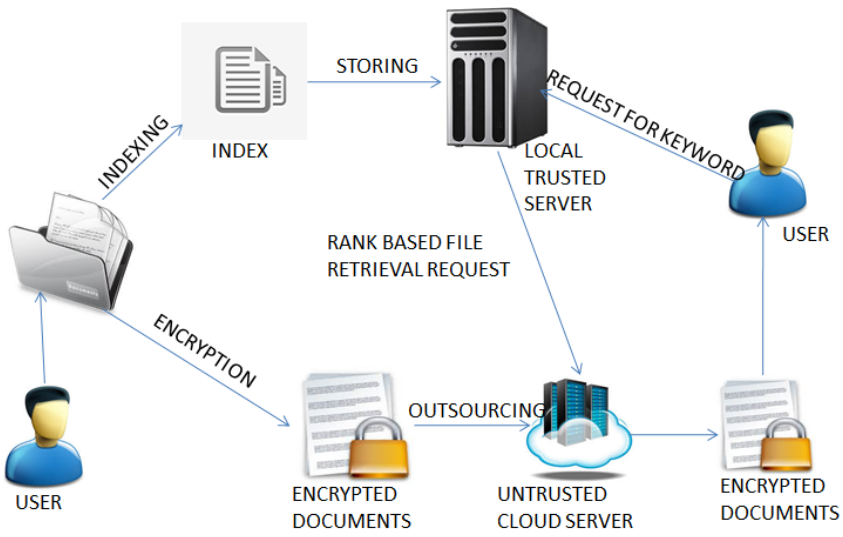

Figure 1.1: Proposed System Architecture

Inverted Index

The basic steps carried out in inverted index are-

1. Collect the documents that are to be indexed.
2. Tokenize the text, turning each document to a set of tokens.

3. Normalize the tokens by either converting all the tokens into lower case or upper case, then perform linguisticpre-processing to the tokens.
4. Index the documents such that for each term occurrence an inverted index is created consisting of a dictionary and posting list.

Each document has a unique serial number known as the document identifier. During index construction, we can assign successive integers to each document when it is first encountered. In this paper, we assume that the first three steps have already been done and the list of normalized tokens for each document has been sent as input for indexing. Now the core step of indexing is to sort the list in alphabetical order. Multiple occurrences of a term in a document are taken as a single entry. Since a term may occur in number of documents we take only a single instance of that term in the dictionary and a posting list is attached to that term which consists of the entire document id's in which the term is present. This posting list is implemented using a linked list[4]. Similar operations are performed on all the terms. Thus, finally we will have a dictionary which will have all the unique terms and a posting list to each term. Inclusion of stop words is optional. In this paper, we have excluded the stop words. Figure 1.2 shows the inverted index generated for the documents given in Table 1.

Table 1: Example Documents with contents

Document Name	Contents
Doc1	Football is famous sport played across the globe.
Doc2	Manchester united and Real Madrid are famous teams in football.
Doc3	Manchester united is one of the famous team in Europe.
Doc4	Football is a team game

Figure 1.2: Inverted Index

Let the data user query be "Famous Football team". Respective posting lists are retrieved for every term in the query as shown in Fig. 1.3

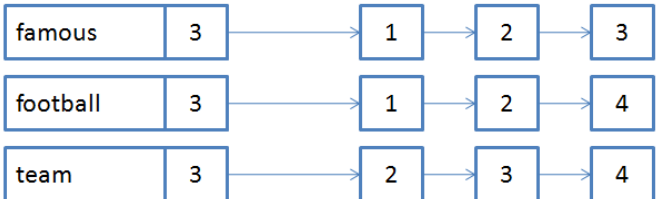

Figure 1.3: Inverted Index for Query

Now, intersections are performed and the resulting document ids are then sent to cloud[8].In this case document id 2 will be sent to the cloud.

Authentication Using DH

Key Agreement is a procedure through which a session key for asymmetric algorithm is generated from the common secret key[1]. The system has two parameters p and g. They both are public and it can be used by all users in a system. The parameter p is a randomly selected prime number of at least 300 digits and parameter g is an integer called as generator. Generally, in DH any number of users can be involved in key exchange [2]. Here, Local Server, User and Admin are involved in this DH key exchange for providing authentication. Admin, Local Server and User generate their private keys a, b and c respectively. For a secure key exchange, much larger values of a, b, c and p are needed. p is a prime number of at least 300 digits and the private keys a, b and c can be 100 digit long. Initially, admin uses his private key to compute g^a and sends the computed value to the Local Server where LocalServer computes $(g^a)^b = g^{ab}$ and it is sent to the User. User computes $(g^{ab})^c = g^{abc}$, which is used as the secret keyby the user. Similarly, Local server computes g^b and sends it to the User where User computes $(g^b)^c = g^{bc}$ and sends it to the admin . Admin computes $(g^{bc})^a = g^{abc}$, which is used as the secret key by the admin. User computes g^c and sends it to admin where admin computes $(g^c)^a = g^{ac}$ andit is sent to the local server. Local server finally computes the secret key $(g^{ac})^b = g^{abc}$ .Thus, if the keys g^{abc}modp, computed by all the three users is same then the authentication for retrieving documents from the cloud will be allowed. In case of any mismatch of the key computed the authentication will be denied. The architecture of DH is shown in Fig. 1.4.

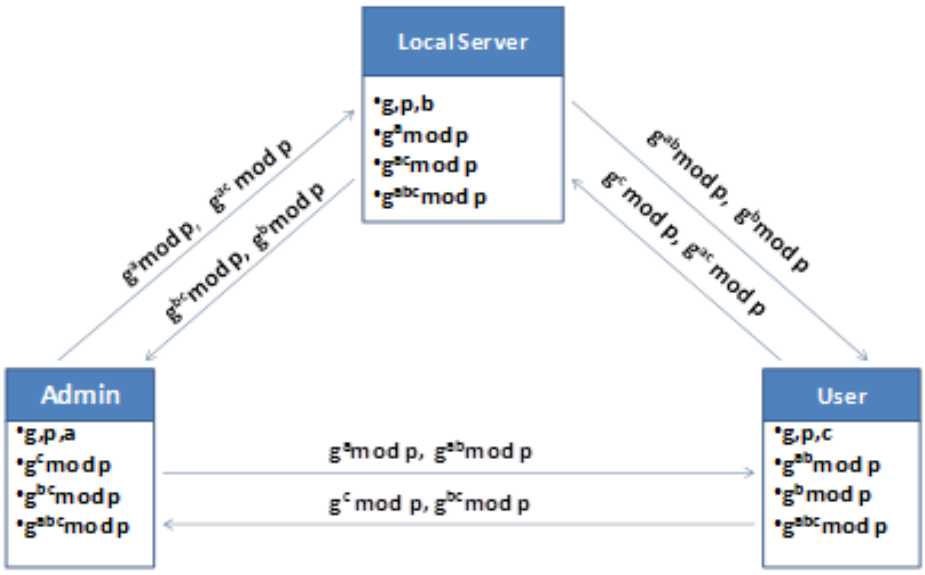

Figure 1.4: Diffie-Hellman Architecture

Elliptical Curve Cryptography:

Encryption:
Based on the time, the algorithms are classified into two types namely polynomial time algorithm and exponential time algorithm. ECC uses concepts of elliptic curves with polynomial time. As ECC follows polynomial time algorithm, the computation is very less compared to exponential time algorithm. ECC has high complexity with fast processing rate.

The elliptic curve equation is given as,

$$y^2 = x^3 + ax + b$$

Both the public key and the private keys are generated through key generation. The sender uses the public key to encrypt the message. Initially the public key K is generated.

$$K = q * P$$

Where, q is a private key ranging from 1 to n-1

P is any point on the elliptic curve.

Let s be the message that has to be sent and S be the point denoting the message s on the curve. C_1 and C_2 are the two cipher texts that has to be generated,

$$C_1 = q * P$$

$$C_2 = S + (q * P)$$

Decryption:
The encrypted message is decrypted to plain text using the private key of the user. The private key is also generated through key generation method. The point S is retained from the cipher text using the following equation,

$$S = C_2 - q * C_1$$

Thus the message s corresponding to the point S on the curve is retrieved.

Ranking
Ranking the document is the next step after storing and indexing the documents, So that based on the ranking the top most relevant documents can be retrieved. Scoring of the document is mandatory for ranking, which is done based on the widely used ranking functions which are based on tf*idf rule,tf is the term frequency representing the number of times a keyword is present in the document, and idf is the Inverse Document Frequency defined as the ratio of number of documents containing the word to the total number of documents present in the server. Tf-idf is used for ranking because it increases the ratio of the term frequency in the document, but counter balances the frequency of the words, so that one can know the fact that general words are more frequent, which are mostly stop words[5].

Ranking Function:
Score (W,Fi) = 1/|Fi|.(1+ln fi,t).(1+N/ft)
W: Keyword score that is to be calculated
fi,t:Frequency of term in file Fi
|Fi|:Length of the file
N:Total number of files in the collection

Conclusion and Future Work

In this paper, we have proposed a design to optimise the search time , ie. the time taken to retrieve the relevant documents that the user needs ,by rank search method, where indexing is done in forehand using IR techniques, thus retrieving relevant documents. Encryption and decryption of documents is also implemented using ECC for efficient storage. We have also provided authentication by using Diffie-Hellman key exchange method. The work can also be extended to handle multiple cloud user requests in single point of time.

References

[1] Preeti, Sharma Bandana," Review Paper on Security in Diffie-Hellman Algorithm," International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 3, March 2014, ISSN: 2277 128X.

[2] S. Anahita Mortazavi, Alireza Nemaney Pour," An Efficient Distributed Group Key Management using Hierarchical Approach with Diffie-Hellman Symmetric Algorithm: DHSA," International symposium on computer networks distributed systems, February 23-24, 2011.

[3] C. Wang, N. Cao, K. Ren, and W. Lou"Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data," IEEE Transac-tions on parallel and distributed systems, vol. 23,no. 8 (2012).

[4] J. Zobel, A. Moat," Inverted documents for Text Search Engines," ACM Computing Surveys, 38(2), 2006.

[5] R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval. Addison Wesley, 1999.

[6] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou"Fuzzy Keyword Search over Encrypted Data in Cloud Computing," in Proc. of IEEE INFOCOM10 Mini-Conference (2010).

[7] M.Venkatesh, M.R.Sumalatha, Mr.C.SelvaKumar,"Improving Public Au-ditability, Data Possession in Data Storage Security for Cloud Comput-ing," In Information Technology (ICRTIT), 2012 International Confer-ence, April 2012.

[8] W N. Cao, C. Wang, M. Li, K. Ren, and. Lou,"Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data," Proc.IEEE INFOCOM 11, 2011.

[9] IR models: Vector Space Model, http://www.csee.umbc.edu/ ian/irF02/ lectures/07Models-VSM.pdf.

[10] L. Harn, W.-J. Hsin M. Mehta, Authenticated DiffieHellman key agreement protocol using a single cryptographic assumption, IEEE Proc.Commun., Vol. 152, No. 4, August 2005.

[11] J. Edge, *An introduction to elliptic curve cryptography*, http://lwn.net/Articles/174127/, 2006.

[12] Enge A. "Elliptic curves and their applications to cryptography", Norwell, MA: Kulwer Academic publishers 1999.

[13] Mohsen Machhout eat.al., coupled FPGA/ASIC implementation of elliptic curve crypto-processor,International Journal of Network Security & its Applications Vol. 2 No. 2 April 2010.

[14] Neha Tirthani, Ganesan R, "Data Security in CloudArchitecture Based on Diffie Hellman and Elliptical CurveCryptography", School of Computing Sciences andEngineering, Chennai

[15] Neil Koblitz, " An Elliptic Curve implementation of the finite field digital signature algorithm", inAdvances in cryptology,(CRYPTO 1998), Springer Lecture Notes in computer science, 1462, 327-337,1998.