An Efficient Encrypted Data Searching Over Out Sourced Data

P.Rama Lakshmi[1], Nagabhushana Rao Chunduru[2], Amarendra Kothalanka[3]
[1]M.Tech scholar,[2] Associate professor,[3]professor
[1,3]Deparment of Computer Science and Engineering in Dadi institute of Engineering and technology.
[2]Department of Information Technology at Dadi Institute of Engineering & Technology.

Abstract: Searching data over Out sourcing is still an important research issue in the field of cloud computing or service oriented application ,because of retrieving the user interesting and relevant results based on the user query. Obviously the out sourced data will be encrypted form. Our approach searches keywords in the encrypted documents in optimal manner based on the file relevance score over service oriented applications

I.INTRODUCTION

Cloud Computing enables cloud customers to remotely store their data into the cloud so as to enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources [1]. The benefits brought by this new computing model include but are not limited to: relief of the burden for storage management and the universal data access with independent geographical locations and the avoidance of capital expenditure on hardware and software and personnel maintenances, etc [2]. With the prevalence of cloud services, more and more sensitive information are being centralized into the cloud servers and such as emails and personal health records etc [3]. To protect data privacy and combat unsolicited accesses and the sensitive data has to be encrypted before outsourcing [4] so as to provide end-to-end data confidentiality assurance in the cloud and beyond. The data encryption makes effective data utilization a very challenging task given that there could be a large amount of outsourced data files. In Cloud Computing the data owners may share their outsourced data with a large number of users and who might want to only retrieve certain specific data files they are interested in during a given session. The most popular ways to do so is through keyword based search. That keyword search technique allows users to selectively retrieve files of interest and has been widely applied in plaintext search scenarios [5]. Data encryption and which restricts user's ability to perform keyword

search and further demands the protection of keyword privacy and that makes the traditional plaintext search methods fail for encrypted cloud data. The traditional searchable encryption schemes (e.g. [6]–[10], to list a few) allow a user to securely search over encrypted data through keywords without first decrypting it and that techniques support only conventional Boolean keyword search1, without capturing any relevance of the files in the search result. When directly applied in large collaborative data outsourcing cloud environment and they may be suffer from the following two main drawbacks. For each search request and the users without pre-knowledge of the encrypted cloud data have to go through every retrieved file in order to find ones most matching their interest and it which demands possibly large amount of post processing overhead; Invariably sending back all files solely based on presence/absence of the keyword further incurs large unnecessary network traffic and which is absolutely undesirable in today's pay-as-you-use cloud paradigm. In short and the lacking of effective mechanisms to ensure the file retrieval accuracy is a significant drawback of existing searchable encryption schemes in the context of Cloud Computing. The state-of-the-art in information retrieval (IR) community has already been utilizing various scoring mechanisms to quantify and rank-order the relevance of files in response to any given search query. The importance of ranked search has received attention for a long history in the context of plaintext searching by IR community and surprisingly. It is still being overlooked and remains to be addressed in the context of encrypted data search.

Design Goals :

To enable ranked searchable symmetric encryption for effective utilization of outsourced cloud data under there mentioned model in our system design should achieve the following security and performance guarantee. Specifically we have the following goals[3]:

i) Ranked keyword search: This search is to explore different mechanisms for designing effective ranked search schemes based on the existing searchable encryption framework;

ii) ii) Security guarantee: to prevent cloud server from learning the plaintext of either the data files or searched keywords and then achieve the as-strong-as-possible security strength compared to the existing searchable encryption schemes;

iii) Efficiency: The explained above goals should be achieved with minimum communication and computation overhead.

II. RELATED WORK

Now a day's secure data storage over cloud servers is an important research issue in the field of cloud computing, Even though various traditional approaches are there for cloud storage, but they are not optimal, because many of the traditional mechanisms are not optimal for data correctness, integrity and dynamic data support.

In Cloud, To ensure users that their data are indeed stored appropriately and kept intact all the time in the service. In this Module Data can be divided in to number of chunks and performs encryption on the block of packets for the security and correctness.

To maintain the same level of storage correctness assurance even if users modify, delete, or append their data files in the cloud. In this module various verification mechanisms are introduced for modify, delete and append operations.

User: an entity, who has data to be stored in the cloud and relies on the cloud for data storage and computation, can be either enterprise or individual customers.

.Cloud Server (CS): an entity, which is managed by cloud Service provider (CSP) to provide data storage service and has significant storage space and computation resources

.

Third-Party Auditor: an optional third party auditor, who has expertise and capabilities that users may not have and it is trusted to assess and expose risk of cloud storage services on behalf of the users upon request.

III.PROPOSED SYSTEM

In this approach data owner out sources the data in the server, before storing data in the server , Data owner has a collection of n data files C = (F_1,F_2, …, F_n) that he wants to outsource on the server in encrypted form while still keeping the capability to search through them for effective data utilization reasons. The data owner will first build a secure searchable index I from a set of m distinct keywords W = (w_1,w_2, …,w_m) extracted from the file collection C and store the index I and the encrypted file collection C on the server. After searching the information data can be organized after the ranking.

Before outsourcing the data owner will first build a secure searchable index I from a set of m distinct keywords W = (w_1,w_2…,w_m) extracted from the file collection C.Index table contains the unique keywords from the datasets along with file ids, before placing them into the index table encrypt the keywords by using symmetric key approach with AES algorithm for security purpose.

Algorithm for index table generation:

Srep1. Read the document F

Step2. Segment the document term wise and encrypt with key

Step3. Calculate term frequency (TF) and inverse document frequency(IDF) and publishing time(P_T)

Step4. Generate index table(I_{table}) and files upload to server

Rijandael algorithm

Rijandael is an efficient and secure cryptographic algorithm regarding computational complexity issues also. Rijandael developed to resolve the drawbacks in DES algorithm. It is a block cipher algorithm.

Key Expansion

Key can be generated by the key schedule of rijandael key schedule. AES requires a separate 128-bit round key block for each round plus one more.

Add Round Key: every individual byte of the state in each byte of the state is integrated with the round key using bitwise xor Rounds

Sub Bytes: By using the lookup table every byte can be replaced by another with respect to the table non-linear substitution way.

Shift Rows: Each row of the block transposed cyclically with certain number of steps.

Mix Columns: Mixing operation which operates on the columns of the state, combining the four bytes in each column.

AddRoundKey

1. Final Round (no MixColumns)
    1. SubBytes
    2. ShiftRows
    3. AddRoundKey

Complete implemention of the subbytes,shiftrows,mix columns and add round key as follows[16], for implementation details we had used builtin algorithm from the dotnet namespaces.

Sub Bytes:

In the SubBytes step, each byte $a_{i,j}$ in the state matrix is replaced with a SubByte $S(a_{i,j})$ using an 8-bit substitution box, theRijndael S-box. This operation provides the non-linearity in the cipher. The S-box is used derived from multiplicative inverse over**GF**(2^8), known to have good non-linearity properties. To reduce the attacks based on simple algebraic properties and the S-box is constructed by combining the inverse function with an invertible transformation. The S-box is chosen to avoid any fixed points (and so is a derangement), i.e.,$S(a_{i,i}) \neq a_{i,i}$, and also any opposite fixed points, i.e., $S(a_{i,j}) \oplus a_{i,j} \neq 0xFF$.

Shift rows:

The Shift Rows step operates on the rows of the state; it cyclically shifts the bytes in each row by a certain offset. For AES first row is left unchanged. Every byte of the second row is shifted one to the left. As same as previous step, the third and fourth rows are shifted by offsets of two and three respectively. For blocks of sizes 128 bits and 192 bits and the shifting pattern is the same. Row n is shifted left circular by n-1 bytes. Similarly each column of the output state of the Shift Rows step is composed of bytes from each column of the input state. For a 256-bit block, the first row is unchanged and the shifting for the second, third and fourth row is 1 byte, 2 bytes and 3 bytes respectively—this change only applies for the Rijndael cipher when used with a 256-bit block as same as AES does not use 256-bit blocks. The importance of this step is making the columns not linear independent and AES becomes four independent block ciphers.

MixColumns step

In the Mix Columns step, the four bytes of each column of the state are combined using an invertible linear transformation. The Mix Columns function takes four bytes as input and outputs four bytes and each input byte affects all four output bytes. Together with Shift Rows, Mix Columns provide a diffusion in the cipher.

During this operation, each column is multiplied by the known matrix that for the 128-bit key is:

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

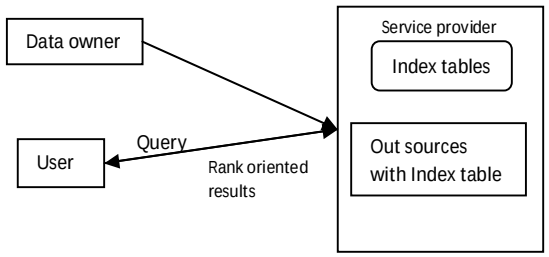
The multiplication operation is defined as: multiplication by 1 means no change and the multiplication by 2 means shifting to the left and the multiplication by 3 means shifting to the left and then performing XOR with the initial unshifted value. After shifting, a conditional XORwith 0x1B should be performed if the shifted value is larger than 0xFF.

In more general sense, each column is treated as a polynomial over **GF**(2^8) and is then multiplied modulo x^4+1 with a fixed polynomial $c(x) = 0x03 \cdot x^3 + x^2 + x + 0x02$. The coefficients are displayed in their hexadecimal equivalent of the binary representation of bit polynomials from **GF**(2)[x]. The Mix Columns step also viewed as multiplication by a particular MDS matrix in a finite field. This method is described further in the article Rijndael mix columns.

AddRoundKey

In the AddRoundKey step, the subkey is combined with the state. For every round a sub key is derived from the main key using Rijndael's key schedule; each subkey is the same size as the state. Sub key is added by combining each byte of the state with the corresponding byte of the subkey using bitwise XOR.

Our proposed Architecture works with web services (service oriented applications) ,that provides the language interoperability and security. The Server receives the query from the user and it encrypts the query by using AES algorithm and authenticates himself with the user key and compares with the encrypted keyword in the index table, finds the number of occurrences of the keyword,



Architechture

that determines the term frequency and inverse document frequency for finding the file relevance score.

In this paper we proposed a novel file relevance score measurement with number of terms in the file, number of occurrences of the term(term frequency) and number of files

relevance_Scores[j] = Convert.ToDecimal((1 / termsinfile[j]) * (1 + Math.Log(termfreqs[j])) * Math.Log(1 + (filecount / numberoffiles)));

Ranking function calculates the term frequency and inverse document frequency for finding the score of the query or keyword with respect to the files, and forwards the datasets according to the score to the user based on ranking.

Architecture Flow:

Files can be retrieved based on the our novel file relevance scores

Step1: Registration of the user at Server by requesting the key

Step2: User receives the key for authenticated and secure search

Step3: User searches for relevant data with a plain keyword

Step4: Service process the query and checks for the authentication of user

Step5 : Service retrieves the relevant information from index table for respective keyword

Step6 : calculates the file relevance scores based on the file relevance score

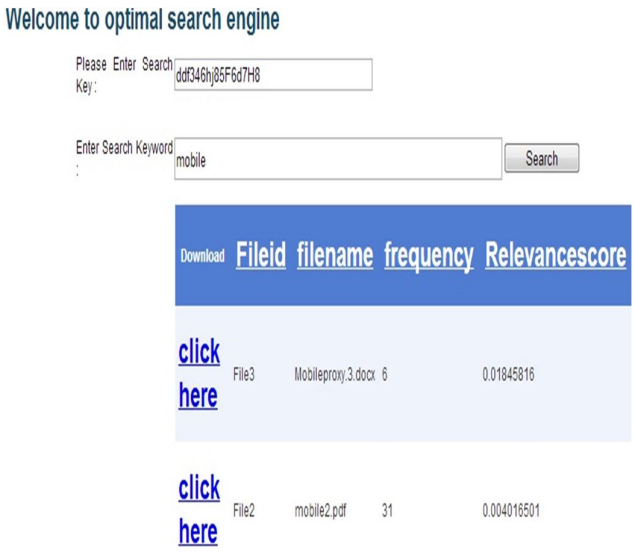
relevance_Scores[j] = Convert.ToDecimal((1 / termsinfile[j]) * (1 + Math.Log(termfreqs[j])) * Math.Log(1 + (filecount / numberoffiles)));

Step7: return the files based on the file relevance score to user

Experimental Analysis:

For implementation purpose we had used C#.net and ASP.net,our experimental analysis shows in the index table generation at the data owners end as follows

keyword	Encryptedword	Frequency
Abstract: In	9ZJppmTvTBHYjA7hnzpZrA==	1
this	fqOkRqkmOb7RAqxWfVNVew==	1
paper	WrhHdUwjmmSxyrsX6bNw4w==	1
we	9nI082yZv8zgzZ+/9OcL+A==	1
propose	UrZJTjmissISF1PcnYZBbg==	1
an	Etn30QcsJ1GfNebLIJ+YfA==	1
architecture	L1/z6C+aL04J1vT6iA02mA==	1
is	It2OxvAvpBC9RTq7p8p1oA==	1
the	19uMT+w1AbRToPI4eFSNZA==	8
design	OlaNjlBwSkI1AXFE0Kmrqw==	1
secure	RLw7uc7+FEMi0294Q7MY9w==	3
auditing	hdNIq/3/B+Iv1q3TnCashg==	2
protocols,	a3EdxNpH98aw1K8Fy0BMPw==	1
during	RkcII4hSTB+/M++DeogzEA==	1
data	1LubZAABBg3ERm0wBW3xmQ==	5
uploading	cUlhqRAWljtZUi7Sm8pT0g==	1
to	WrO3Vk5qMQylESeYvBsWQw==	2
server	Uo80X6N1mqCWTDKJ+WKZnQ==	2



IV.CONCLUSION

Our approach provides an efficient secure search mechanism over service oriented application with relevant files by calculating the file relevance scores of the files which contains the search keyword, encrypting the keyword at server side and retrieves the relevant information.

REFERENCES

1. B. Bloom, \Space/time trade-o®s in hash coding with allowable errors," in Communications of the ACM, Vol. 13(7), pp. 422{426, 1970.

2. M. Bellare, R. Canetti, and H. Krawczyk, \Keying hash functions for message au thentication," in Proceedings of CRYPTO'96, Lecture Notes in Computer Science 1109, pp. 1{15.

3. D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano, \Public key encryption with keyword search," in Proceedings of Eurocrypt 2004, Lecture Notes in Computer Science 3027, pp. 506{522.

4. K. Bennett, C. Grotho®, T. Horozov, and I. Patrascu, \E±cient sharing of encrypted data," in Proceedings of ACISP 2002, Lecture Notes in Computer Science 2384, pp. 107{120.

5. O. Goldreich, Foundations of Cryptography: Basic Tools, Cambridge University Press, 2001.

[6] B. Krebs, "Payment Processor Breach May Be Largest Ever," Online at http://voices.washingtonpost.com/securityfix/2009/01/ payment processor breach may b.html, Jan. 2009.

[7] I. H. Witten, A. Moffat, and T. C. Bell, "Managing gigabytes: Compressing and indexing documents and images," Morgan Kaufmann Publishing, San Francisco, May 1999.

[8] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proc. of IEEE Symposium on Security and Privacy'00, 2000.
[9] E.-J. Goh, "Secure indexes," Cryptology ePrint Archive, Report 2003/216, 2003, http://eprint.iacr.org/.

[10] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in Proc. of EUROCRYP'04, volume 3027 of LNCS. Springer, 2004.
[11]http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

BIOGRAPHIES

P.Rama Lakshmi completed her btech in JNTU ,Now pursuing M.tech in Department of Computer Science and Engineering in Dadi institute of Engineering and technology. Her Interested areas are Data mining and network security.

Nagabhushana Rao Chunduru completed his M.Tech in Computer Science & Technology from Andhra University and completed M.Sc in Computer Science from Andhra University. He has Ten years experience in teaching. Currently he is working as Associate professor in the Department of Information Technology at Dadi Institute of Engineering & Technology. His interested areas are Data Warehousing & Data Mining, Computer Networks.

Amarendra Kothalanka is a Professor & Head of the Department of Computer Science & Engineering, Dadi Institute of Engineering and Technology (Affiliated to JNTUK), Anakapalle, Andhra Pradesh, India. He obtained his M.Tech. in Computer Science & Technology from Andhra University. He is pursuing his Ph.D in Computer Science & Engineering from GITAM University,Visakhapatnam. His main research interests are Safety Critical Computer Systems, Software Engineering and Mobile Computing. He is the Sponsor of DIET ACM Student, Women in Computing and Professional Chapters.